M Sasikumar

Introduction to Artificial Intelligence

Buzzwords of Artificial Intelligence, Expert Systems, etc are found often in the literature. There are also cousins of this including neural networks, genetic algorithms, etc. As with many terms in the IT industry, there is no common minimum definition for the term AI. When philosophers are still debating what intelligence is, we can not expect to put a precise definition for the artificial counterpart of it. People in various disciplines, attaches different meanings to this term. Some see it as an attempt to create a new species which will be superior to humans. Some see it as a means to understand human intelligence and mind. Yet others see it as a natural stage of evolution in computing technology in becoming more and more powerful and less and less intimidating to use.

In this chapter, we will attempt to sketch a picture of this vague territory. We will not attempt to be exhaustive or precise. The goal is to give you a feel for what AI means, its motivation and driving forces, differences from conventional (the non-AI) programming, and take a peep into some of the practical applications.

We will assume that you have some general understanding of computers. You are also familiar with the typical applications of computers – such as word processing, accounting, data processing, etc.

The following quotation from Newell, gives us an idea of the fundamental difference in viewpoints between the AI researchers and the other computer science researchers. ``*The digital computer field defined computers as machines that manipulated numbers. The great thing was, adherents said, that everything could be encoded into numbers, even instructions. In contrast, the scientists in AI saw computers as machines that manipulated symbols. The great thing was, they said, that everything could be encoded into symbols, even numbers.*''

In other words, the foundation of conventional programming is the notion that everything, including text, graphics, images, and instructions, can be numerically coded and hence can be manipulated in a machine. The numerical encoding was of paramount importance. However, in AI the numerical values are of significance. We are interested in higher level pictures of these. For example, a system trying to recognise digits from an image has no use for the numerical value or representation of digits. The physical appearance is the critical factor there. The discussion below will help to exemplify this notion further.

What is AI?

The first application of computers was in military during the world war. The task: to decypher coded messages. These were largely numeric applications – trying out various combinations of coding. Once the war was over, the machines were put to other applications. Much of the earlier applications were in payroll processing, numerical analysis, etc. Today, we have computers being used for word processing, animation, and a wide variety of applications.

If you examine these applications, there are some restrictions which are taken for granted in almost all of them. They are all characterised by fixed pre-planned sequence of instructions. At any point, you know exactly what to do. And hence you are able to program the machine to execute those specific instructions at specific points. For example, we can pin point the actions to be taken if a user clicks the left-mouse button when he is in a certain menu on the screen. This is not to imply that the task of doing that action is trivial. But it is possible to identify the action. Our notion of computer programming has been based largely on concepts like flow charts – the decision points in the chart identify changes in the path being taken. All points where a change is likely to happen are identified and all possible paths are enumerated.

On the other hand, consider some of the more challenging applications we process every day. We largely follow preplanned sequences – atleast after one has got trained in them. We don’t think about much of our regular actions such as walking, talking etc. When we leave from home to work, we visualise a typical day. But when we reach the bus-stop we notice that the regular bus had an accident and the next bus will be too late. We never had an action menu at this point. But now, we need to create a new menu and then choose something from it. We create a list of potential actions on the spot. It is unlikely that we visualised all these possibilities before leaving from home.

On another front, consider the task of a doctor. When a patient comes with a set of symptoms, in the general case, it is impossible to know precisely what is the problem. Based on the available information, which may be quite inadequate, he is forced to make some decisions about the possible disease. Two days later, when the patient returns he may find out, that his decision was wrong! So he corrects the path. When you are going to a new place and loose the way, you can come back along the path you took and then try another path. Only time is lost. But when a doctor has to “undo” his

previous decision, it may not be easy. The effects of the previous decision could be anything – it may have to be corrected (if it was not too fatal!).

These illustrate a different class of applications, which does not yield to simple programming. Now computers are beginning to be applied to the solution of such “complex” tasks. What are some of the characteristics of these complex tasks? We saw some of them in the simple examples above. Let us look at the list in a little bit more systematically.

No single visible path to solution: When you take a step now, it is not possible to know which direction that step will take you. Locally, it may look good – but a few steps later you may be in trouble. For instance, some medicines will provide apparent improvement in the short run so that everyone believes that the treatment has been effective. But a week later, suddenly complications emerge! Similarly, an apparent degradation can be followed by a substantial improvement. Another example would be stock markets! Note: I will keep using the medical domain as an example, since the concepts will be familiar to most people – not because of any hard feelings against the doctors!

Paths of ‘no return’: This is related to the previous point. In some cases the lack of visibility can reach an extreme. If the decision turns out to be fatal, there can not be reversal of the decision. If in Chess, you make a move and the opponent declares win, you cannot undo and say, “sorry, that is not what I wanted to do”.

Cost of different options: In general, there is more than one option open at many points in these tasks. The different possibilites may yield to different outcomes as we mentioned above. In addition, they may also cost differently. For example, when you need to go home from work, you can take a taxi every day – it is a workable solution. But it may cost you more money than what we can afford. So, in general, we would need some other solution. But there may be circumstances when a more costly solution is ideal – such as when a delay in reaching could cost you an important contract or miss a date!

Uncertainty: Many domains have inherent uncertainties. Often we dont think of them consciously. When we talk of outside being “hot”, or a person being “tall”, we are dealing with one kind of uncertainty. At what point does the description short give way to tall in describing heights of people. What happens when a specific piece of information that is requested is not available at present? What happens if your sources of information are not completely reliable (deceptive sources or simply faulty instruments)?

Rapidly changing situation and environment: In applications like that

of monitoring, autonomous navigation, etc the environment is continuously changing. Parameter values are valid only for limited periods of time. And therefore decisions that were taken are also valid only for limited period of time. If a temperature reading today indicates fever, the patient is not expected to be under fever for ever.

Now the question in front of us is “Can we handle such tasks”? If we are solving a task which shares some of these characteristics we will term such a system as intelligent or knowledge based.

This is a quote attributed to Dijsktra - ``*The effort of using machines to mimic the human mind has always struck me as rather silly. I would rather use them to mimic something better.*'' That is another way of describing what we are looking for. We are not interested in a singing, dancing, crying robot. Human mind is amazing in many ways, but has its own share of limitations also. So if we are going to imitate something, why should we target human mind for it? Anyway much of the interest in AI for computer professionals is from the concern of solving more complex problems, or making the machines more productive and usable machines.

Applications

Now let us take a look at a few applications in a little more details to see where the complexity comes from. This will also tell us something about the application areas which AI researchers are targetting at.

We do not aim to sample all areas of AI here. As a matter, we will concentrate on some of the areas that NCST has been active in. These are machine translation, intelligent computer aided instruction and heuristic scheduling.

Machine Translation

When we talk comfortably in two different languages, or translate sentences from one language to another, the task seems fairly easy to do. But this is something that has been bothering AI researchers for a few decades now. An automated approach to translation from one language to another has enormous scope of application.

One easy to spot example is the coverage of news items in our regional language news papers. Most of the news is reported in English. Covering them in regional language requires translating them to local languages. Being news items, translations have to be done relatively fast. A comparison of the size of news papers in English and any of the regional languages will

give you a reasonable picture of the complexity of this task.

In countries such as India having a number of regional dialects, ensuring that information reaches all parts of the population is a hard task just due to the amount of translation effort required. A fancier application would be something like an international phone. I could call up someone in Japan and talk in Hindi. When he hears my voice in his receiver, it would be in Japanese. Similarly he would reply in Japanese and I would hear him in Hindi! Here, we are talking of not just automated translation, but real-time translation as well as speech recognition and synthesis.

Another application is information retrieval. Todays document retrieval systems are based on keywords. The systems do not try to understand the text of the document to determine relevance to a user query. This model does meet much of our requirements, but can miss relevant documents because the document used a synonym of the word. For example, if you search for documents related to Rajiv Gandhi, we would consider anything talking about Indian Prime Minister during a specific time period as also relevant. These notions of dependencies stretch limitlessly.

Though Machine translation or language understanding has many applications or high value, there are few systems in the market today for this task. The reason is just the complexity of the task. To get a feel of the complexity of machine translation, attempt translating these into your favourite language.

- He came by car.
- He came by night.
- He came by the town.
- He came by 3 pm.

These are very simple sentences by any standard. The point to note here is that the preposition by will map into different constructs in the target language, though the use of by in all the four sentences are quite similar in structure. The only difference is in the meaning of the noun following the preposition. So a translation system should not only know that car, night etc are nouns, but much more: for example, that night refers to a time period, car is a vehicle and hence a mode of transport. Even for simple sentences like this, we need a lot of knowledge about the world to generate correct translations. The next pair of sentences indicates a similar, but different situation.

John opened the door.

The key opened the door.

Despite the structural similarity of the sentences, one does not visualise Mr key walking up to the door and pushing it open, nor John being inserted in the keyhole to open the lock. Now consider the following pair, again having a similar structure.

- I fell in love.
- I fell in water.

Here the issue is handling of proverbs and language specific phrases. This requires more special handling to recognise such usages. In the target there may not be a similar phrase at all and if there is a similar usage it may not be in the same structural form.

The examples above are meant to give you a feel of the complexity of the problem. As they show, it is not enough to know the syntax of the two languages. Even for fairly simple sentences, lots of world knowledge may be required. The peculiarities of the language such as phrases, idioms, etc are also important to know in order to carry out translation of text properly. When we discuss Natural Language Processing later in the text (Chapter ~\ref{nlp}), we will look at these issues and how AI attempts to address them in more detail.

Tutoring Systems

With increasing accessibility to computers, information technology is beginning to play a significant role in education. Computer based tutoring packages have been around for many years now. One of the earliest in this respect is the learn package on Unix. They are essentially page turners, organised as a fixed series of lessons, each of which would consist of a piece of text, with or without graphics etc, to be presented to the user to read and understand.

If the system includes an assessment component, it will typically consist of a small number of canned questions based on the material covered in that part. If the user obtains a score above a pre-specified threshold, he is allowed to proceed to the next lesson. Otherwise, the current material is re-presented to him for a second go. There is no attempt to understand the users difficulties and redress the same. Whatever was the reason the user did not manage to clear the test, the remedy is to go through the entire material again. For a human teacher, the type of errors that the user

makes, and the questions answered correctly/wrongly are all valuable indications of what the student has learned, more importantly of what he has not learned. His remedial procedure will consist of providing more material or more practice emphasising the specific weaknesses as could be detected from the analysis of the answers.

This ability to model the students understanding and provide material and tests according to the student model would make the tutoring systems more effective. This is the domain of Intelligent Tutoring systems. Modelling a student is not the only thing characterising good teaching. What material to present, at what stage, in what manner are what distinguishes a good teacher from others. Powerful representations of the domain structure and dependencies of the various components inside are important for this.

In an era of shortage of good and qualified teachers, this is an important technology to be exploited. More so, where education is no longer something you do get a job, but something you need to sustain till the end of one's life.

Resource Scheduling

Now let us move to another class of applications. This is an area familiar to computer programmers and researchers for long. The domain is resource scheduling in other words, scheduling allocation of resources to meet specific requirements while minimising the total cost. This is a vast domain covering a number of types of problems. Some examples are:

- Scheduling of airline crew (pilot, air-hostesses, etc) to meet flight requirements keeping in mind duty time constraints, crew preferences, availability, etc.
- Planning of bus routes and frequency to meet passenger demand in a city (there is also a related, but different problem of deciding which bus will ply on which route on a specific day).
- Scheduling of vehicles to distribute finished products from a company to its various depots, as well as gathering of relevant raw material from various sources to the company.
- Scheduling of machines for various pending jobs, in a factory.
- Classroom scheduling and examination scheduling in colleges and universities.

As mentioned above, from the early days of computer applications, these problems have received the attention of computer people. For example,

most books on computer algorithms discuss miniature versions of job shop scheduling. A number of mathematical models such as Dynamic Programming, Linear Programming, Graph Algorithms, etc exist for various types of problems. Unfortunately most of these are inadequate to model the complexity of these problems in reality. Finding the best schedule for most of these problems is practically impossible, given the large search space and the complex set of constraints.

Even finding a good schedule that does not violate any of the hard constraints in the domain is a complex task. To illustrate this point, let us consider the classroom scheduling problem. It may appear to be simple.

But consider the kind of real-life constraints that could arise:

- Teachers – subject preference (typically a many to many relation).
- Students – subject mapping, again a many to many relation. If a student is taking two subjects, the two subjects should not be scheduled in parallel sessions.
- Special requirements for certain subjects, such as 2 hours of physics require lab, atleast two consecutive sessions for maths, etc.
- Availability of teachers, classroom and any other infrastructure required. Teachers may be available only at certain times – in these days, many institutions reply on visiting faculty for running their courses and they have other constraints/commitments.
- Classrooms have physical limits on seating capacity. Some subjects may require large classrooms, or a classroom with audio-visual facility, etc.

In addition, there are other preferential constraints such as spread the sessions of each subject across all days of the week, spread teachers load across the available days, etc. You can see that it is not easy to formulate a solution which can take care of all these aspects, and handle realistic problem sizes. In practice, a solution satisfying all constraints may not exist – it may be required to relax one or more constraints.

Identifying the most appropriate constraints to relax adds another dimension of complexity in such problems.

A Few Other Applications

The discussion so far was meant to give you a feel for why Ailike sophistication is required for some of the applications. AI research has addressed problems of a wide variety of domains and complexity ranging

from language to robotics and vision. Let us take a quick look at a few other important application areas concerned with AI.

We will be discussing these in detail in later chapters of the book. It is useful to keep these problems at the back of your mind, as you read through the general material on AI in the early chapters of the book.

Planning: This is a vast domain concerned with sequencing of operations for achieving a goal, arrangement of items according to specifications, assigning time and resources for a set of operations, etc. These kinds of tasks are known by different names, in general. For example, configuration refers to arrangement of a set of given items to meet some requirement. Scheduling brings in the time assignment issue. However, all of these are considered as part of the general planning domain. We will cover planning in more detail in Chapter ~\ref{plan}.

Machine Learning: This term is generally used to refer to systems whose performance improves automatically with time. Depending on the domain and type of representation, a number of sub-fields of this exist today. Learning is also classified as supervised, unsupervised and semi-supervised depending on the amount of feedback available from the environment to the learning system during the learning process.

Neural network based systems have shown the most amount of success in this area. Rule induction has become a practical solution to partially derive domain knowledge from case records. The recent wave of interest in Data mining also derives much from this sub-discipline. The interest in data mining is to examine a large database to derive useful generalisations from it. For example, transaction record of a bank coupled with customer profiles can provide clues about loyal customers and customers likely to be interested in specific schemes that the bank is offering.

Computer vision: Identifying objects from the visual image is often a trivial task for humans to perform, but something that has baffled computer programmers. Computer visions primary focus is to understand an image captured by a camera for example, can you find how many boxes are in the picture, is there a house, how many people are there, etc. Apart from its obvious military applications, it has tremendous industrial applications.

Speech Recognition: Just like vision, most people have little trouble understanding others' speech. Unlike text representations, the speech is often broken, not properly punctuated, ungrammatical, etc. Pronunciations vary from person to person even time to time! Speech recognition also turns out to be a hard task for computers to come to grips with. Recognition systems with limited vocabulary and which are heavily speaker dependent

are today available in the market. But a general purpose system is still a dream!

Intelligent Interfaces: User interfaces have come a long way from the days of text-based terminals. Graphical, what-you-see-is-what-you-get (WYSIWYG) interfaces are common today. You also begin to see more intelligence in the interface to understand the users intention. Multimodal interfaces reduce the need to remember a number of cryptic commands. For example, dragging an icon to the recycle-bin deletes whatever is represented by the icon it could be a file, or directory or whatever. Similarly a double-click opens any icon the system invokes the appropriate program depending on the icon: Text editor, Excel, Web browser, etc. More and more intelligence is required to make computers a tool for everyone. Being able to understand short natural language sentences and queries would be a step. If it could remember and analyse my profile, it could offer more useful help for my work reminding me of proper names of people I interact with, inserting their addresses in letters I make, identifying which database I am referring to (without my having to remember their names, or SQL formats). The field of intelligent agents promises a lot in these areas – creating software programs which can interact with the rest of the world for us negotiating deals, identifying useful information, etc.

These are all areas of interest to AI community. As you can see, they span a very vast territory.

AI Approaches and Paradigms

The above discussion showed a vast range of problems characterised by some of the types of complexity, outlined earlier. They have all been playgrounds for serious research work in AI. A number of practical systems are available in many of these areas, though the intended level of maturity and features are still years away. Similarly a number of techniques and tools have also emerged which are today routinely used for solving practical complex problems in many areas. Based on these problem samples, let us try to define AI and characterise its core components.

As mentioned earlier, the basic problem in all these tasks is that there is no way to know if a step further from where we are, in any direction will get us to the solution or not. If we are at a junction from where we dont know which road will lead to the place we want to reach and if there is no one in the vicinity to ask for directions, we would go and try one of the ways. Either we get to the solution, or we will realise there is no solution in that way and we decide to return to the junction to try the other way. If we are

not sure of the way to a place, we need explore the possible options. However, pure exploration with no information to help us will be very expensive.

Imagine a Martian inhabitant, landing up somewhere in India and trying to make its way to your institute, when the only information it has is your institute name. It will need to wander along all possible roads for arbitrary length, till it finds a building whose name board matches the name you have given. If it had some information that the institute is towards the south of where it has landed, it saves a lot of unnecessary exploration in other areas. If it is further told that it is within 10 kilometers of the place of landing, it can terminate moving along any road beyond 10 kilometers. In the other extreme, if you had given it a series of instruction (turn south, walk 18 meters, turn left 45 degree, walk 32 meters, and so on), it needs essentially no exploration. This is what our normal computer algorithms are like. In other words, the more the information relevant to the solution of the problem you have, the less the amount of exploration required.

[xxx figure]

The situation with AI is quite similar. In all the domains we outlined earlier in this Chapter, we lack complete information, unlike in the case of a program computing salary or aligning text in a word processor. So we will need some trial and error exploration. But we are not totally clueless. In the case of machine translation, we have constraints telling us that ‘eat’ requires an ‘edible’ entity as its object. We can use such ‘knowledge’ to reduce the blind exploration. Recall from your algorithm classes, that most interesting problems in life are of exponential complexity. So enabling us to solve realistic problems in acceptable time and resource bounds require intelligent use of knowledge to the best extent possible. This is what characterises an AI approach to problem solving in general.

Thus, using AI technique involves mapping the problem to a search problem, incrementally moving from one state to another, having a smart algorithm that can search through such a space without wasting time, and an ability to represent and use available domain knowledge to guide the search.

The notion of search space is, therefore, central to AI. In the next chapter, we will study this in detail and see how to view search as a general problem solving methodology. A variety of search techniques have been invented in the course of AI history including those built by borrowing ideas from other disciplines and natural processes. The most generic and simplest are breadth first and depth first search used for any arbitrary graphs. These are

extended to make use of heuristics to guide the search, resulting in models like best first search, A* algorithm, iterated deepening, simulated anhealing, hill climbing, and their many variations.

The second major concern of AI is ways of finding and using knowledge effectively, the problem of Knowledge Representation. After search, we will focus on this problem and outline a variety of frameworks for representing and using different types of knowledge for different requirements. The simplest type is the familiar ‘if-then rules’: ‘if you have yellow colouration in your eyes and if you have been having fever for a few days, then you are likely to be having jaundice’. Other models like include frames and semantic networks (precursors to the object oriented programming popular today), constraints, and the formal mechanism of logic. One major aspect of knowledge representation is capturing and representing uncertainty. In the rule above we used the construct ‘likely to be having’. How does one represent this? How does one represent notions like “he is a *good* student”, “that is an *old* story”, etc. We look at models such as bayesian networks, certainty factors, fuzzy logic, etc.

After this, we will look at various sub-fields of AI, mostly focussing specific application or problems such as machine learning, planning, natural language processing, etc.

Before we conclude this intro, let us take a brief look at the history of the field as well as some applications which are in practical use.

A Peek into the History

Most authors consider the Dartmouth conference in 1957 as the origin of the field AI. The phrase ‘artificial intelligence’ was coined by John McCarthy during the conference. It brought together many of the early pioneers of AI including Marvin Minsky, Herbert Simon, and Allen Newell. The next few years saw phenomenal growth in the field, though the computers in those days were primitive compared to the machines of today.

The early attention for the AI enthusiasts fell on games. Playing games like Chess, Go, Tic-tac-toe, etc requires good degree of human intelligence. Therefore, if we could program machines to play these games well enough, then we are close enough to mimicing human intelligence, so went the rationale. Simple models of problem solving as in the case of GPS (General Problem Solver) formulated by Newell and Simon, did surprisingly well with simple instances of games. One coded rules of the form, ‘if you are in state X and you want to reach Y, then try this operator/transformation A’.

These simple rules provided enough capability to produce programs capable of playing games like tic-tac-toe well.

The idea was extended to simple planning problems such as the monkey and banana, blocks world, etc. They worked. It looked as if one could solve any problem by just representing the relevant knowledge in terms of these rules. Researchers went overboard making statements such as computers will beat chess world champion in 10 years time.

As the platform for experiment moved to games like Chess as well as the apparently simpler problems of natural language processing, vision, etc, the wind changed direction. Beyond toy problems, the model was not scalable. Time went by; none of the predictions showed any signs of materialising. Governments reviewed research work in AI; massive cuts in funding followed. AI carried a major negative connotation. US and UK governments set the trend in motion. This was the beginning of a period, now called ‘AI Winter’.

It took a few years for AI to rediscover itself – but this time with more realistic targets, more along the lines we outlined in the beginning of this chapter. Rather than attempt to address human-level of intelligence in general, the focus shifted to formulating techniques for solving specific problems. The economical success of the field expert systems brought a great level of credibility back to AI. Other demonstrable systems followed in areas such as planning, speech, vision, etc. Today, AI is once again an exciting and supported area of research the world over.

In parallel with these developments, the field of Neural networks also went through a similar behaviour pattern. The perceptron model evoked a lot of interest and expectations. The availability of an automated learning algorithm was one major factor in this excitement. A number of similar models proliferated – Adaline, for example.

At this time, Minsky took to study the Perceptron formally. Papert and Minsky published their results in the now famous book “Perceptron”, which literally killed the field of neural networks for over 20 years. Their study showed that Perceptron system is of little practical value, since it cannot even learn the simple XOR pattern. At the end of the study, there was a caveat that these results may not hold for multilayer perceptron networks – but nobody bothered to follow up!

The development of the backpropagation training algorithm on multi-layer perceptron brought back the attention to neural works. Today, it is an exciting area of research with a number of different models and vast range of applications.

Today one can find a large number of conferences, international and national, dedicated to various subfields of AI, and a few focussing on AI in general. AAAI, ECAI and IJCAI are examples of the later. A number of journals also focus on AI and related disciplines. All these signify the increasing interest in the various aspects of AI the world over.

Conclusion

In this note, we have tried to give you a feel for the challenges motivating the field of Artificial Intelligence and for the various components that, today, make up the field.

One question that many AI students have is the programming language they should use for doing AI. A few years back, this was an important question. The high level languages of those days, were too low level for AI programs. Languages like Lisp, Prolog, etc were almost necessary for achieving any meaningful AI behaviour in a course project; otherwise, most of the time will be spent in working out lower level issues such as memory management and flow control, to get any time for the actual problem. Today, we have quite powerful languages and powerful machines.

Languages such as C++ and Java can be used comfortably for working out AI programs. A number of libraries are available in many languages which will help you get started quickly.

Further Reading

There is a special conference called “Innovative Applications of AI” (IAAI) held every year which showcases interesting new applications of AI techniques. These cover a mix of commercial and academic applications. AI Magazine often publishes special issues soon after the conference, focussing a select set of these proceedings.

http://www.aaai.org is the official site from the American Association of Artificial Intelligence. They feature an excellent portal on AI. On almost any topic of AI, you can find applications, tutorials, seminal paper, etc. In addition, there is a good section on AI news and AI applications. It is certainly worth a bit of your time, if you are seriously interested in AI.

=+=