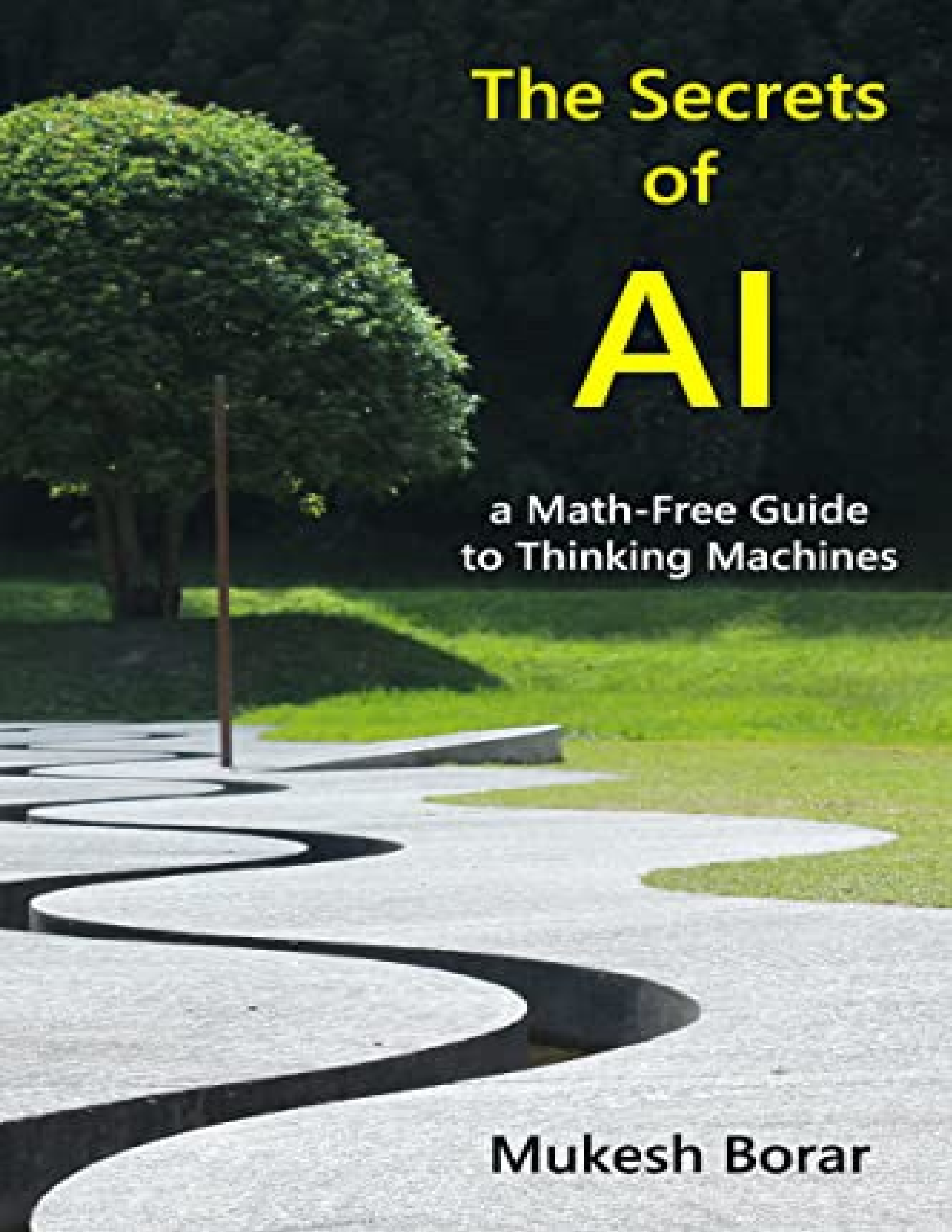


The Secrets of AI

a Math-Free Guide to Thinking Machines

Mukesh Borar

THE SECRETS OF AI

a Math-Free Guide to Thinking Machines

Mukesh Borar

CONTENTS

PREFACE

Math and magic are two words that can easily be associated with AI. How machines talk, listen, read, compose, observe, and act is nothing short of magic. For many tasks, machines already perform better than the experts in the field. A lot of this is happening because machines can learn, and they are getting better at it with each passing day. Machines learn simply by observing massive amount of data. At times, this appears magical even to those who are from the AI fraternity.

Often the magic stems from the sophisticated mathematics that powers the machine learning algorithms. The math involved can range from basic arithmetic to advanced concepts like calculus. It includes advanced techniques from the disciplines of statistics and probability. The mystery of AI deepens with the usage of inscrutable neural networks, which are learned through complex mathematical operations.

However, we cannot just live in awe of AI; we must understand it. AI is mastering more and more human activities. It is dramatically expanding the role of computers in our lives. Decisions made by the thinking machines already touch millions in their daily lives. Today, it is a must to have a good appreciation of what goes on behind the scenes. AI knowledge is critical, not just for software professionals but also for those who approve it, invest in it, and use it.

It is here that math can become a hindrance. Acquiring deep knowledge of mathematics is not everyone's cup of tea. This, at times, is true even for software professionals. Math can surely come in the way of developing a good understanding of AI. It can inhibit us from having an informed opinion on AI. That's not how it should be. Math need not be a pre-requisite to know more about AI. AI is too important a technology to not learn about for want of mathematical prowess.

"A math-free language is the best way to reveal the secrets of AI."

This book uses natural language to discuss the fundamentals of AI. It presents a holistic view of AI without being overwhelmed by the cryptic details of learning algorithms. AI is explored from multiple perspectives that are critical to the success of AI programs. Intelligence, design, engineering and trust aspects are covered in detail, without referring to math.

Building an AI-powered system is not just a complex exercise; it is unpredictable as well. Even the seasoned software professionals often find it tough to grasp how AI works. Software organizations frequently struggle with AI programs. *Most organizations do not invent new learning algorithms, do not design new kinds of AI models, and do not synthesize new types of data. Yet, so many AI programs fail.* The failure rate for in-house programs is even higher.

Often, what software organizations need is an integrated view of intelligence, data and the learning process. This would help them make better decisions about the techno-financial feasibility of an AI program. They must also appreciate the additional demands that AI puts on most of the conventional roles. The activities & responsibilities of architects, engineers, analysts and testers need to be oriented towards AI engineering. This book delves deep into all these topics, and reveals the relevant success secrets.

The appreciation of a concept or idea increases significantly when it is illustrated with relevant examples. Recognizing this, the book includes a comprehensive case study – an AI system that takes live coverage of sports to every court, field, ground and stadium. The objective is to cover every sports event live, whether big or small, professional or amateur. This kind of scale is possible only with automated cameras that are controlled through AI. It requires an AI system that chooses right camera angles, shows appropriate replays and creates live-feed without any human intervention. The case study discusses the details of building such a comprehensive AI-powered system. It illustrates the domain study, architecture, engineering and business aspects of developing such a system. This case study reinforces the principles and concepts discussed throughout this book. It also presents several heuristics for AI engineering.

AI is a wonderful technology that is accelerating innovation and creating new classes of systems. It is adding intelligence to almost everything around us. For software professionals, it is opening up new lucrative business avenues and careers. Great fortunes can be made by embracing it in the right domain, and for the right tasks. Now is the best time to create truly intelligent solutions, services and products. This book aims to help you excel at AI, and make the best of this grand opportunity.

1 INTRODUCTION

Since the time of their inception, computers have an ambitious goal to accomplish:

"Master all human tasks that require intelligence and thinking capabilities."

Computers have been steadily marching towards this goal. In the last few decades, we have seen a phenomenal rise in their capabilities for a variety of intelligent tasks. They are now used to transfer money, conduct meetings, optimize traffic, schedule trains, track parcels, run factories and search the web. Their sphere of physical presence is also increasing by the day. They are no longer limited to servers, desktops, tablets and phones. Microprocessors are now embedded in cars, televisions, watches, washing machines, cameras, lights and many more devices. They convert these products into smart devices with a certain degree of thinking capabilities. Computers have a simple but special property, which lets them accomplish these feats: *they are programmable machines.*

Computers can be programmed to do a variety of intelligent tasks. At the very core, a microprocessor can perform just some elementary arithmetic and logical operations. It is the beauty of programming that we can create massive systems through complex combinations of these basic computations. That's what the field of software engineering is all about. Today's large software systems comprise millions of lines of code. We have armies of software engineers in the world who write, maintain and improve these programs. *They encode intelligence into computers and innumerable other devices that have embedded microprocessors.* There is a whole industry that creates tools to assist them. They are well supported by a conceptual backbone of engineering methodologies, programming paradigms and architectural frameworks.

The advances in conventional programming have achieved a lot. Yet there are many tasks that are simply out of reach for these programs. For instance, it is almost impossible for us to write a conventional program, which can identify a human face in a picture. Or, one that can translate a sentence from Spanish to Japanese. The same goes for making a robot walk, or making a car navigate through a complex situation on the road. Over seven decades of programming experience does not give us any indication that conventional programming would succeed in mastering these tasks. We need alternative methods to program the computers for such activities.

These are the kinds of tasks that computers can master through AI. AI is exceptionally good for many of the jobs that are out of reach for conventional programming. *AI takes computer programs beyond the realm of traditional manually written code.* It opens up a world of computer programs that are created through machine learning. Most of these programs are learned by processing and observing data, often very large quantity of data. Through learning algorithms, computers extract intelligence from training data and store it in AI models. Perhaps that is the reason for this intelligence to be called as 'artificial'. *It is 'artificial' as it is extracted by machines, and not encoded by software engineers.*

AI dramatically expands the scope of computer programs. The intelligence that was out of reach for conventional programs can now be made available to computers. Machine learning truly marks an important step in computers' journey to master all human intelligence. Computers can now perform many more jobs and activities. The following diagram depicts this development:

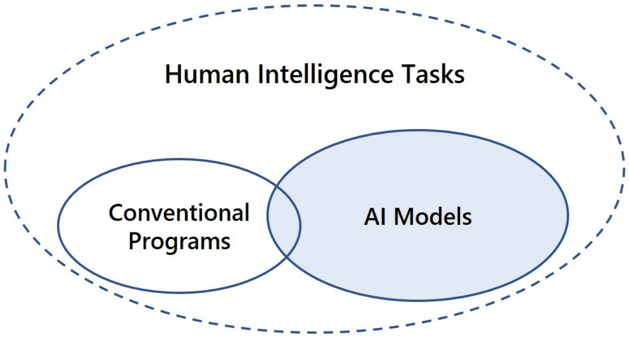


Figure 1: The expanded scope of computer programs

This diagram also brings up a few important questions to the fore. Can AI models, which are the executable AI programs, replace conventional programs? Can AI models also accomplish what conventional programs do well? Can we reduce conventional programming and rely largely on machine learning? The answer to these questions is, by and large, negative. *AI models are not replacing conventional programs.* It is much easier to program a computer to transfer money than to make it learn the same from data. The same goes for tasks like booking flight tickets, scheduling production in a factory, managing inventory and tracking orders. In fact, this is the case for almost all the processes and procedures that have been designed by humans. *Programmability seems to be an inherent ingredient of all the business processes that humans design.* This is true despite the fact that business processes predate computers and programming. Even centuries old processes can be represented using flowcharts and process models – the tools that are often used to guide programming.

"The processes, policies and procedures designed by humans, are best encoded by conventional programming. AI is not changing that."

There does exist a small overlap between conventional and AI programs. It especially exists where conventional programs implement the analytics models. The specifications of these manually written models are usually derived from statistical analysis of data. AI models can compete with conventional programs in such areas. However, this is just a small overlap.

The Role of an AI Engineer

Machines do not have inherent capabilities to learn by observing the data. They still perform just the same elementary arithmetic and logical operations, which are well exploited by conventional programs. To learn from data, they need learning algorithms and a few more artifacts. These artifacts are designed and developed by AI engineers. The role of an engineer is as significant for AI systems as it is for conventional ones.

"Machines learn, but not on their own. They learn under

comprehensive guidance of AI engineers.”

However, it takes a different kind of engineering perspective to make machines learn. *It requires a mental shift from programming to learning.* And that is not a small shift. Software professionals have to acquire and practice a new thought process to make machines learn. The entire development lifecycle has to be adapted towards the nuances of machine learning.

Still, AI engineering is not dramatically separated from conventional software engineering. Rather, it builds on many of the foundations laid by conventional programming. Software engineers need new but highly related skills to take on the role of AI engineers. AI provides a huge new opportunity for both software professionals and the software development organizations. This book derives its inspiration from this opportunity.

The Structure of the Book

This book is organized in two parts. Part I covers the fundamental concepts of AI. Part II discusses the engineering heuristics for building AI-powered systems.

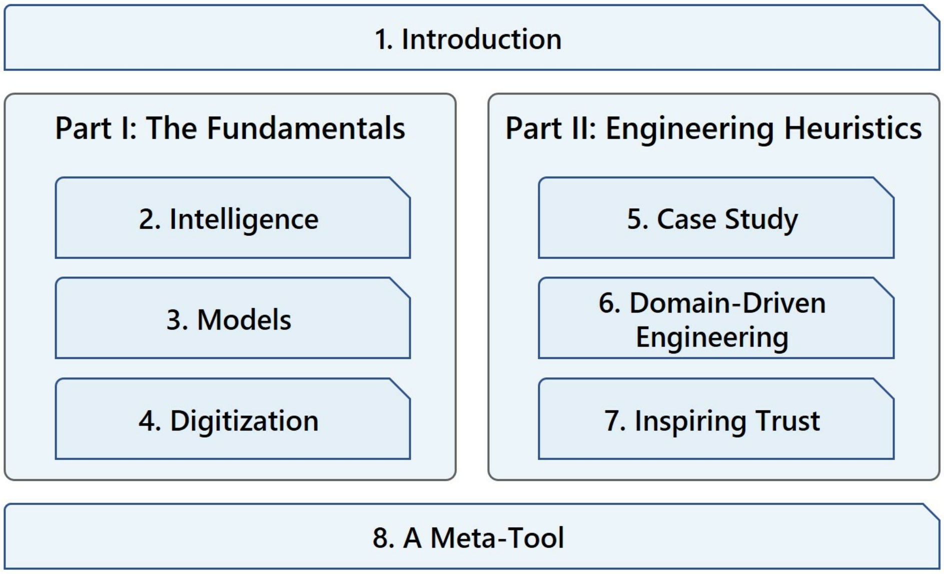


Figure 2: The chapters of the book

The fundamentals-part begins with a holistic view of multiple kinds of intelligence that AI is destined to master. Chapter 2 describes the intelligence characteristics that are critical to spot the business opportunities, appreciate the complexity of AI systems, assess the risks, and plan for success. It presents a conceptual framework for AI, which is used throughout the book to reveal AI success secrets.

Models are at the core of AI. Good models are essential for AI's success and growth. Chapter 3 describes various classes of model architectures and how they represent intelligence. It unravels the exceptional power that neural networks have for capturing diverse kinds of intelligence.

Digitization forms the foundation for AI. It is crucial to have a good digital representation of a domain for the machines to learn about it. Ironically, digitization enables as well as constrains machine learning. The efficacy of machine learning often depends on the efficiency of digitization. Chapter 4 explores different approaches to digitization, and how they make the real world accessible to machines.

The engineering part starts with an exhaustive case study that illustrates the AI development lifecycle. Nuances of domain study, digitization, labeling, architecture, and model definitions are discussed in detail. Heuristics like leveraging prior learnings through enriched features, are described with examples. The case study in Chapter 5 also includes the challenges faced after the deployment of AI systems. It provides heuristics to overcome them and build the system for diverse conditions.

The importance of domain knowledge for AI engineering cannot be overemphasized. Chapter 6 presents a domain-driven approach to developing AI-powered systems. It covers heuristics for data coverage, feature enrichment and model architectures.

Trust in AI is a complex issue with a number of factors working against it. Many of the issues are almost intractable for end users. Even for creators of the system there are significant trust related challenges. At times, the proposed solutions like explain-ability are of little use and may even add to distrust. Chapter 7 explores the inherent causes of distrust, and also the steps that can inspire trust in AI models.

AI is set to universalize the access to intellectual expertise of most kinds. This would include scientific, medical, social, legal and financial expertise. The concluding chapter, Chapter 8, explores the role of AI as a meta-tool. It describes how AI is acquiring a place of prominence in our general intellectual space.

2 ANATOMY OF INTELLIGENCE

The landscape of human skills, acumen, know-how, ingenuity and expertise is huge. The canvas of intelligence seems infinite. There are many hues and colors to human genius. They range from extreme dexterity in certain jobs to a riot of creativity leading to innovative solutions. New products, recipes, and gadgets are created even by people with no formal education in the domain. Almost every day, we see examples of individual brilliance that celebrates vastness of human intelligence.

There is another critical aspect of human intelligence, which is as important as individual ingenuity. It is based on the fact that we humans are social beings. We share a lot with our fellow beings, and that includes intelligence. There is a good degree of commonality in the skills, know-how and expertise that we all possess. For example, we have a common understanding of traffic signs, ATM functions, time tables, flight check-in process, icons in popular mobile apps, and so on. While it may not be obvious, the shared understanding also applies to more complex tasks, like driving and production planning. Indeed, the intelligence and knowledge required for most of our routine activities, are universal.

The ‘universality of intelligence’ is a concept of great significance for AI. It tells us about the potential lifespan and utility of an AI model. It shapes the architecture of an AI-powered system, impacts AI engineering, and determines the economics of training data. It helps us in identifying and assessing the business opportunities. As we will see in this chapter and the rest of the book, this concept is at the heart of the exponential growth of AI. In the next section, we explore this concept further before moving on to its larger implications for AI.

Universality of Intelligence

The universality of intelligence means that we have a shared impression of the underlying intelligence. However, it does not mean that everyone has all the knowledge and skills. The commonality of intelligence and knowledge is limited to a group of people. It is the ‘group’ that has the shared understanding. The ‘group’ here is defined by the task or the activity being considered. When examining an X-ray film for a possible fracture, the group comprises radiologists and other doctors. Given an image, they all are likely to point out a fracture correctly. When it is about flying an aircraft, the group would comprise pilots trained for that kind of aircraft. When the task is to detect a human face in a picture, most people in the world would have the required intelligence. The concept of ‘universality of intelligence’ holds irrespective of the size of the group.

The group itself is dynamic. New radiologists, doctors, engineers, pilots, drivers, firemen, etc. keep joining their respective groups. And, they keep imbibing the same intelligence, knowledge and skills. A new doctor develops the same skills, to spot a fracture in X-ray image, which the experienced doctors already have. A new pilot learns the same flying maneuvers that the existing pilots already know. It is the beauty of universality of intelligence that the concept holds while the group keeps evolving. It works because the underlying intelligence remains same and shareable.

How do we all end up sharing so much common knowledge, skills and expertise? We owe it to a few of our innate skills. Our inherent ability to learn is harnessed towards this purpose. Our sensory intelligence, the ability to make sense of what we see, hear and touch, contributes actively. The society capitalizes on these innate qualities to impart us common knowledge and skills. A child starts imbibing common intelligence at home, which is reinforced and extended by the society in general.

Teaching is a formalized method to promote this cause. A large part of teaching is about imparting common knowledge to pupils. Things do not stop with formal education. Workplace training usually imparts the knowledge of a standardized way of working. Trainings often provide information on best practices. The practices that are to be followed by the team, for good results and to avoid errors. Standard Operating Procedures (SOP) are designed where there can be multiple diverse ways to handle a situation. These are the tools to ensure commonality of action. With SOPs, people know not only about how to act but also what others may do in a particular situation.

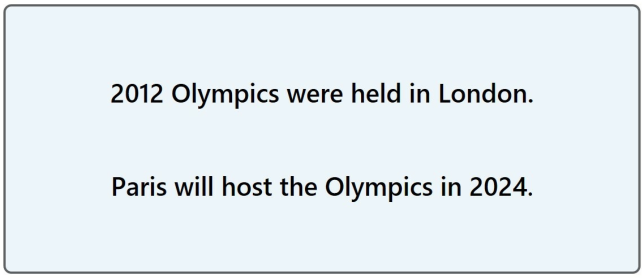
The shared appreciation of knowledge, skills and expertise, is the cornerstone of human progress. It is the basis for humans to work in teams and form larger societies. The universality of intelligence enables cooperation within and across teams of people. It is the foundation of inter-dependent life that spans from small families and villages to large cities and nations.

“The universal nature of intelligence is the driving force behind most human interactions.”

While society contributes significantly toward this cause, there are two fundamental factors that ensure the universality of intelligence. They are the static nature of intelligence and shared mental patterns. The former is an inherent property of intelligence, especially of the intelligence required for routine tasks. The latter refers to the common imprints of intelligence patterns that are formed in our brains. They both have profound implications for AI, which we explore in the next two sections.

Shared and Static

The intelligence required for most of our routine activities is not just shared, it is quite static as well. It does not change much with time or even place. Perhaps the intelligence becomes universal only because it is static and stable. The static nature provides for the time required for it to be shared within a group of people, even when the group itself is continuously evolving. It allows time for the universalization through techniques like teaching and training. To explore this aspect of intelligence, let’s have a look at the two sentences below:



2012 Olympics were held in London.

Paris will host the Olympics in 2024.

Figure 3: Two sentences about the Olympics

Almost everyone with a reasonable knowledge of English language would develop similar understandings from these sentences. It does not matter which part of the world one lives in, the meaning derived would be quite the same. If one were to read the first sentence in the year 2010, one would conclude that it is phrased incorrectly. Yet the association derived between ‘London’ and ‘2012 Olympics’ will be same as that derived by someone reading it in 2015. If someone were to read the second sentence in 2026, the conclusion would be that it was written before the said Olympics were held. In short, *the meanings of sentences transcend time and space.* That’s why we are able to make sense of the books written hundreds of years ago, by authors who lived thousands of miles away. That’s how the words engraved on stones can tell us the stories of the times they were written in.

The intelligence or knowledge required to understand a language is fairly static. It does not change much with time. Change, if any, is quite slow; it can take years and decades. This has important implications for AI models. A good AI model that translates English into Japanese can transcend the boundaries of time. A language translation model developed in 2020 would be equally useful in 2024. Of course, we cannot go back in time, but if we could, it would have been useful in 2010 as well.

Mental Patterns

Our mind forms certain mental patterns for our learnings, skills and knowledge. These mental patterns represent our intelligence and know-how. When intelligence is shared within a group, the mental patterns formed in the minds of group members tend to be similar. We can get a glimpse of this by having a look at the following logical reasoning exercises:

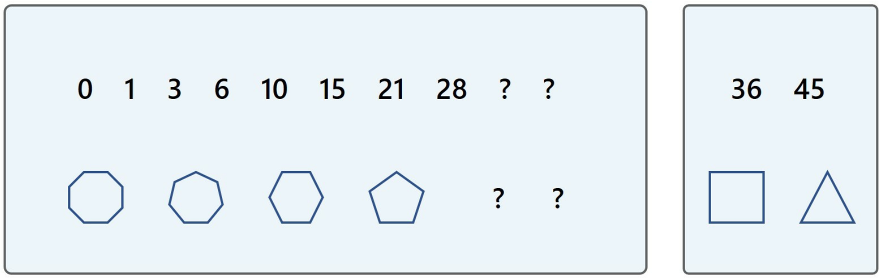


Figure 4: Logical reasoning exercises

There is a very good chance that most people would reach the solutions that are given on the right side. How does this happen? How is it that so many people reach the same solution? It happens because we have developed a common understanding of the logic involved. *We have common imprints of the concerned mental patterns.* They help us reach the same conclusion quickly. Such common imprints are not limited to logical reasoning. They apply to other universal intelligence tasks as well. Let us have a look at the two pictures below:

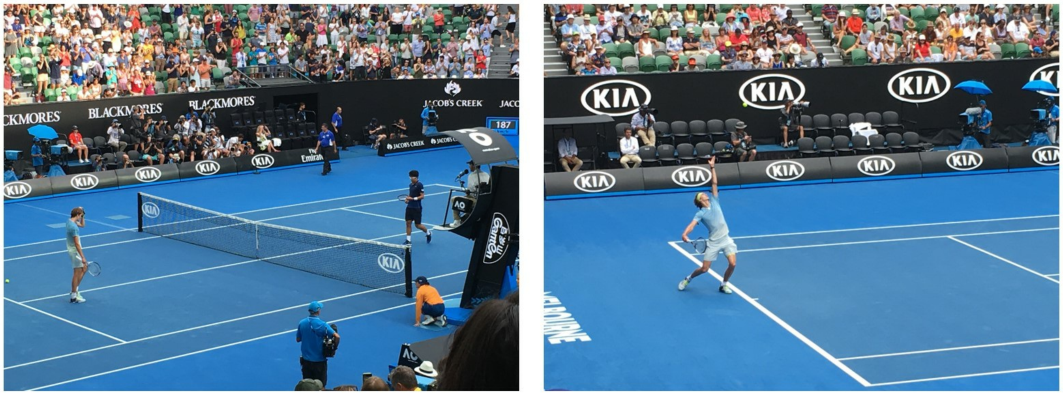


Figure 5: Two pictures of a sports event

Can you identify the sport from these two pictures? If you know about tennis, it should be easy to answer. However, if you did not know about it, you would normally say ‘I don’t know’. Now think of a game that you know well, say badminton (or volleyball, or basketball). Can you say the pictures above are of badminton? Whether you know about tennis or not, you are unlikely to say badminton for these pictures. The reason is simple. In our minds, the tennis pictures would never map to a well-known badminton. Only some sort of confusion or illusion can lead to such mapping. That’s how strong the mental patterns are. It works in this manner for most of us, because we all have almost the same mental patterns in our minds.

Can we make a wild guess when we do not know much about tennis? Yes, we can, but for that we have to have only a vague idea of the guessed sport. Let’s say we have heard about racquetball but have never seen it or read much about it. We can then make a wild guess of racquetball for tennis pictures, because we do not have the mental patterns for racquetball to block this guess.

AI is about mastering these mental patterns. Machine learning works by learning some sort of digital representation of these mental models. An AI model can recognize the sport from a picture when it has learned to imitate the concerned mental pattern. An AI model can translate from German to English when it has imbibed our translation related mental patterns.

How many patterns should AI master before it can do majority of human tasks, especially the routine ones? This is a tough question to answer, but we do have a major hint. The universality of intelligence indicates that there are common mental patterns for most of our routine activities. It also indicates that these patterns are both strong and stable. This means:

> *“Machines trying to master human tasks are not faced with an infinite variety of intelligence patterns to learn about.”*

This is not to say that human ingenuity is bounded or stagnant. It simply means that AI can be very successful by mastering a finite number of intelligence patterns. *Great progress can be made just by focusing on key elementary mental patterns.* We will see an illustration of this statement in the case study covered in Chapter 5.

A Promise for AI

“Write Once, Run Anywhere” (WORA) was a slogan used in the 90s for Java programming language. There was a promise that the code written in Java would run on many platforms, with zero modifications. A Java program

could be written on any device, compiled into a standard bytecode, and would then run on any hardware that had a Java Virtual Machine implementation. WORA heralded a new era of code portability. It was one of the reasons for Java to become such a popular programming language.

In the field of AI, the concept of ‘universality of intelligence’ is the basis for a similar sounding statement that makes a bigger promise:

“Learn Once, Use Forever”

We humans do this all the times. We learn things once and can use it anytime and just about anywhere in the world. We learn a language and are able to use it for our lifetime. When we learn to recognize various objects, the knowledge tends to stay with us forever. Though we may get a bit out of practice, we rarely forget cycling or swimming even after years of not indulging in the activity. We learn the principles of engineering, and simply keep using them in our engineering work, no matter where we are working. The list just goes on, and the tasks can be from diverse domains like sports, economics, technology, accounting and more.

Thanks to the ‘universality of intelligence’, AI can easily follow this for a large number of tasks. ‘Learn Once, Use Forever’ has deep implications for AI models. It indicates that a well-trained AI model can have both longevity and wide applicability. For example, once a robot learns to walk on certain kinds of surfaces, its trained AI model can be used for years. It can also be used by any number of similar robots. The bots that clean floors in houses would do it in any number of houses, day in day out, and for many years to come. X-ray machines can include a model that would mark the potential fractures in the image. This model can easily be used for the lifetime of the machine; and these machines can be used in different geographical locations across the globe.

The ‘Learn Once’ part of the promise does not mean that learning stops after an attempt is successful. Just like we humans can get better at language or driving skills, so can AI. There will definitely be new improved AI models even for the tasks that adhere to the principle of universality. Improving things is in the very nature of human activity. We keep creating better wheels despite the proverb *‘don’t reinvent the wheel’*. In fact, reinventing is a major opportunity avenue for innovators. So often, innovations are new improved ways to address previously solved problems. Yes, we will find new ways and architectures to create superior AI models. Yet a new model is unlikely to affect the utility and applicability of an existing model. This is akin to the utility of an existing phone model when a new improved one is introduced in the market. The new model may make the older one obsolete but would not affect its utility.

The ’Use Forever’ part has huge financial significance for AI models. When something has longevity, a lot more can be invested to build it than for something that has a shorter lifespan. This is one of the reasons for so much of investment that is going into building of language processing models. These language models have both wide applicability and longevity. They can be provided ‘as a service’ for a long time to come, and to a large number of users. The concept of Universality of Intelligence is indeed a major factor in the investment decisions for AI systems.

A Champion of Reuse

From a software engineering perspective, *the universality of intelligence champions the cause of reusability*. An AI model that captures a universal intelligence pattern can be widely reused. It can also be used as a building block in the larger systems. Thanks to universality of intelligence, the developers of AI systems can leverage the AI models that were designed and trained by others. Let’s consider the case of a new camera company called 3D-Cam. It requires a face detection model to support the focusing function of its cameras. It decides to design and train a new face detection AI model. This would require the collection of a fairly large amount of data from diverse sources, and labeling it appropriately. It would also require multiple rounds of training, testing and architecture refinement. All these are expensive activities, and there is a good chance that success may still elude the company. Is there an alternative? There is. Since the intelligence required for face detection is universal, the task of face detection is same whether it is required by 3D-Cam or any other camera company. 3D-Cam can easily reuse pre-trained AI models, which are already being used by others. This entails several benefits. 3D-Cam gets to use a proven well-designed model, whose performance is well established.

“Reusable, pre-trained models are set to rule the AI world.”

Pre-trained models are going to be a major segment in the business of AI. In many areas they are already well-established. When a bank wants to automate the reading of documents, it does not invest in developing a new character recognition model. It would normally pick up one that is available off-the-shelf. The ‘universality of intelligence’ is a major factor that influences ‘build vs buy’ decisions. The reuse is promoted not just through off-the-shelf reuse but also through retraining a pre-trained model with new proprietary data. The benefits also accrue through the reuse of the architecture of successful models. We discuss such reusability in more detail in chapters 5 and 6.

Dynamic Knowledge

The world around us is not a static place. There is a lot that goes on all the time. Sometimes there are patterns in what is going on, and other times they are just random occurrences. There is good value in deriving insights by observing what is going on. Businesses can benefit from knowledge of the latest trends and state of affairs. There is value in detecting patterns that may exist. Most of these patterns are dynamic. They change with time, sometimes quickly, like in the case of traffic, and sometimes slowly, like people’s liking for a particular holiday resort.

A recommendation engine is a good example of a model that captures dynamic knowledge. Recommendation engines are used on most of the popular e-commerce platforms. These engines tell us about the products that were bought along with the one we are viewing. Like, people who bought a tennis racquet, also bought a cap. This knowledge of product P1 being bought with P2 is dynamic. It is about people’s behavior, which varies with time. It varies with changing preferences, evolving needs, prior purchases, occasions, events, and many more factors. There are long term trends, seasonal variations and even daily fluctuations. How do the platforms gain this knowledge? They gain it through statistical analysis of data. They process the available transaction data. Given the ever-changing nature of this data, this statistical processing is not a one-time task. It has to be repeated periodically to be in sync with customer behavior.

“Learn Once, Update Regularly”

The AI models that store dynamic knowledge need to be updated regularly with new training data. We have to find a balance between historical and recent data, for their continuous training. It shall be noted that the dynamism is due to changes in data and not so much in the way we learn from it. For this reason, the models are conceptually stable. Once we have created a good model, we know when to conclude that product P1 is bought with P2. The logic required to do so does not change all the time. We just need the right mix of recent and historical data to draw the conclusions.

The dynamic knowledge that is gained through the statistical analysis of data is usually not of universal kind. A group of people cannot meaningfully have a common understanding of it. The reasons for this include the following:

1. Statistically learned knowledge is heavily dependent on the sample data that it is derived from. It is not easy for all the group members to refer to the same data all the time. When the samples differ, so would the inferences, even when every group member uses the same inference model.
2. It is tough for all the group members to remain updated as and when the new data arrives. Members can easily get out of sync with the current situation.

Technology is helping us in addressing these issues. Machines can easily capture the dynamic knowledge. Statistical analysis is their forte. Computers are good at storing and processing large amount of data. More so when the data is structured, like the sales data on an e-commerce platform. Since this knowledge is time-sensitive, machines need to constantly process the most recent and relevant data. That too is a strong area for machines. Aided by multiple technologies and increasing computing power, machines can easily collect and process the most relevant data. Things are getting better as most of the transactions are moving online. The Internet of Things (IoT) is connecting more and more devices. Over time, the availability of real time data would only increase. We would not only learn from the latest data but also make the inferences available to all interested people in real time.

"Providing universal access to dynamic knowledge is a major AI business opportunity."

Real-time data capture, processing and dissemination would significantly universalize the dynamic knowledge. We would be able to apply dynamic knowledge to real life situations, and that too in a consistent manner across numerous situations. This is already happening. Today, traffic conditions are monitored in real time through smart phones of subscribers. Well-designed AI models are used to compute best routes for commuters. This way, the relevant dynamic knowledge is made available to stakeholders, in real time. It provides community level benefits.

Design Intelligence

"You can analyze the past, but you need to design the future." This is a quote from Dr Edward De Bono, a well-known authority on thinking skills. It says quite a lot about analysis and design. One looks at the past, and the other intends to shape the future. With analysis skills we take a deeper look at the world around us. We analyze it, we discover why it is the way it is, and we develop insights into its workings. With design we change the world, we contribute towards general progress, and we transform things for better. We can analyze the current state we are in, but to make progress we need design skills. Indeed, design is a special thinking skill that we humans possess. It shows us a way forward from current situation.

For centuries we have known that "change is the only constant in life". When it comes to change through innovations, our design skills are the primary contributor. There is a massive amount of design activity going on around us all the time. We humans just can't stop designing new things. Design is our inherent and integral trait. Each one of us is blessed with primary design skills, which we keep refining throughout our lives. Our design activity is a major reason for the perennial change around us. This collective activity of ours keeps changing the world through new ideas, products, policies, services and solutions.

Design is usually a two-stage process, which can be summarized in three words:

“Generate Evaluate Iterate”

Most new designs start with generation of ideas that have the potential to achieve the design goals. The ideas contribute towards solutions. They evolve into solutions through the application of our design skills, aided by heuristics, patterns and templates. Our knowledge of the domain and prior design experiences make major contributions. A design is usually proposed towards certain goals and objectives. It needs to be assessed against these objectives. This involves reviews, tests, prototypes, proofs of concept, and more. At times, there is also a need to check for the compliance with various standards, regulations and guidelines. In many domains there are well laid out processes, rules and metrics that help us with assessments.

Generate is the creative stage of a design process while the Evaluate stage is analytical. The entire process iterates between the two. The switch can happen in a matter of minutes and seconds. There are no demarcated boundaries between the two, and it may appear that the process is in both stages simultaneously. When large teams are involved, things may be more formalized, and the two stages may be better separated. In large teams, each iteration of Generate and Evaluate stages can take weeks to months. However, the brains of individuals do not stick to just one of them for a long time. In fact, in his book Lateral Thinking, Dr De Bono emphasizes on the need to suspend assessment to allow an idea to live longer. This is proposed as a way to not kill a nascent idea, and allow it to contribute better towards the design. However, to do so one needs a lot of conscious practice. Generate and Evaluate tend to be parallel activities in our minds.

How universal are our design skills? Is design a skill that is shared within a group of people? Would a group of interior designers come up with same designs for a house? Would two teams of architects design a new stadium in the same manner? The answer to these two questions is a clear no. Design is an act of creativity. It cannot be same across a group of people. Even if a group follows fairly standardized processes, each member is likely to come up with a different design for a given context.

Given this, it may come as a surprise that people can cohesively work in large design teams. Teams comprising hundreds of people, routinely work to design a new car or a building, a space station or a software system. The reason why it works is – the appreciation of design is quite universal. People in a specialized group have a common understanding on evaluating a design. There are standardized processes and metrics to review the designs. More so when we are assessing the design on its measurable objectives. The evaluation of design does fall under the category of intelligence that is shared within a group of people. Even for the Generate part, people may refer to common templates and patterns. Often the solutions are reused for similar contexts, sometimes with minor modifications. Reuse saves cost, time and effort. It is a major design goal, especially in large organizations. In some industries, it has benefits that go well beyond design stage. The benefits of reuse extend to manufacturing, maintenance and repairs as well. Templates, patterns, standards and reuse, all combine to bring in significant degree of universality to our design skills. They form a base on which large teams can work together, and come up with great solutions.

Design AI, or AI for design activities, is a promising area of AI. Glimpses of its potential can be seen in AI models that learn to play games. They learn by generating moves and evaluating them through reward mechanisms. More advanced design capabilities would require higher degree of creativity than just enumerating computable options. Further, there has to be integration with real world. Many designs need to move out of digital space and be evaluated in the real world. New frameworks have to evolve to facilitate the iterations of such creative generate and multi-stage evaluate steps. Simultaneously, new methods have to be designed to let machines be more creative. In many domains we are making huge progress with generative AI. However, many of these generative AI models are likely to be task specific. Given the vast canvas of design activities, we may need new classes of model for many of the design tasks. AI Models for Evaluate part are likely to be more generic. They would have several components that can capitalize on universal knowledge. Given its immense potential and impact, the following can definitely be said about Design AI.

“Exceptional power of AI will be felt when it starts performing design tasks at a scale not imagined today.”

Choice Intelligence

Imagine a common scenario that happens with teams in many workplaces. You have a team of eight people. Recently, your team successfully completed a project, and now is the time to celebrate. You decide to go to a restaurant to have dinner together. The question is, which one to go to? There are multiple ideas, and the team is unable to zero in on one option. Then in a flash of creativity, you come up with a name, and wow, six out of eight people happily agree, and the other two do not have much objection. How did it happen? Maybe it was just luck, or maybe it was your knowledge of what the team generally likes. Maybe most people like the place but had not been there in a while and saw this as a good opportunity. The reasons can be many, and it may be quite tough to zero in on just one.

Does this happen only within a team, or does the same happen with an individual as well? Often, we are faced with multiple options and have to choose one. Whether it turns out to be a good choice or not, we still have to choose. *This can be termed as ‘Choice Intelligence’*. Most of us have to use our choice-skills multiple times on a daily basis. It can be a matter of selecting a restaurant for lunch, or selecting a destination for our next vacation. It could be about taking the morning flight or the evening flight. It may be a trivial matter like picking up a food item from the menu, or deciding on which shirt to wear to the office. Or, it can be a serious matter like which one of the three job offers to accept. We make choices all the time. There is an intelligence required, but we cannot always explain the choices. Even when we have a good reason, it need not be the only one or even the dominant one. The best part is, we do not have to be consistent with our choices. In fact, in so many cases, we have to choose in a manner that is not consistent.

Making choices is like design and yet different. There are goals but they keep changing. At times, breaking monotony is itself a big goal. This means that a current decision is not disconnected with past or the anticipated future.

Can machines make choices for us? If yes, should they be restricted to decisions with little consequence, like which restaurant to have dinner at? Can they also make big ticket decision like accepting a job offer? AI is set to do this for us in more and more areas. Sometimes we may not even be aware that someone else, especially a machine, has made the choice for us. They do it all the times in search results of all kinds. Some results are explicitly marked as sponsored links, but other results too are a product of some algorithm, policy and the underlying data. They are decided by machines for us, and they definitely narrow down our choices.

Choice intelligence is a special area of AI. It needs to learn from data, and it also needs to use some policy guidance. We have to figure out which data to learn from, and how to blend the design goals with these learnings. In a sense, *AI for choice intelligence would combine statistical knowledge with programmable policies and some generative creativity*. It is an area that has huge potential to influence our daily lives. It also entails a number of trust issues, which we will explore in Chapter 7.

Summary

Machines are not faced with infinite variety of intelligence patterns to learn about. A significant share of human skills can be mastered by a finite number of AI models.

Thanks to the universality of intelligence, there is a great business opportunity in developing pre-trained models for a wide variety of tasks.

AI would be instrumental in providing near-universal access to dynamic knowledge.

Design AI is a promising class of AI, which would thrive on innovative integration between AI models and real-world artifacts.

Choice AI is an interesting blend of learnings from data and the rules that are derived from design goals.

3 MODELS: THE ARTIFICIAL BRAINS

The conventional programming paradigm that powers most of the software systems in the world is well summarized by the following commonly used diagram:

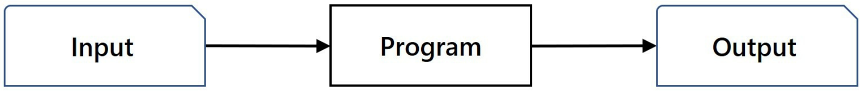


Figure 6: A common representation of conventional programming

A program runs on a computer, it accepts input data and produces the output. These programs are conceptualized, designed and developed by humans; by software engineers to be more specific. Engineers study the input, understand the desired output, and create a machine executable code that converts the input into output. This code can range from a few lines for a simple system to millions of lines for the more advanced ones. It may comprise simple calculations or complex multi-layered logic. Elaborate algorithms are routinely used to accomplish the higher order tasks. The beauty of the above diagram is in its simplicity. It captures the essence of most of the conventional software systems, irrespective of their size and complexity.

Things change a lot when it comes to modern AI that is powered by machine learning. The program, which converts the input into output, is replaced by an AI Model. An AI model is an executable program that runs on a computer just like a conventional program. For a computer, it is not different from a conventional program. It follows the same format, structure, rules and constraints. Yet it is so different from a conventional program that it signifies a paradigm shift. This change is presented in the following diagram:

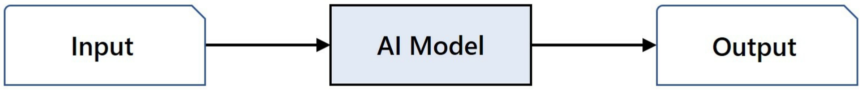


Figure 7: AI model as an executable program

It is a paradigm shift, because unlike conventional programs, an AI model is not written by software engineers. Rather, computers create it, through a process aptly named as machine learning. The computers figure out the details of an AI model, seemingly just by observing the relevant data. That’s not all; these AI models accomplish the kinds of tasks that are tough or even impossible to achieve with conventional programs. Imagine writing a program to detect human faces in a multi-megapixel image. It is almost impossible to do so. Yet, almost every camera today has this kind of program, and it is machine learned.

Most common way of creating an AI model is to learn it from meticulously collated training data. This training data is quite different from the input of a conventional program. Each unit of the training data is a pair, comprising a conventional input and the associated desired output. The input units are described in terms of a number of features. The output unit is usually a label that is assigned to the input unit. The output unit can also be described by a set of features, just like the input unit. This happens often for generative tasks like translation from one language to another. The learning algorithm goes through these pairs, and figures out the general relationship between input features and the expected output, or output features. More precisely, a learning algorithm figures out a map between the input features and the expected output. At the end of the learning process, this map is stored in the AI model. The following diagram provides an overview of this process of machine learning:

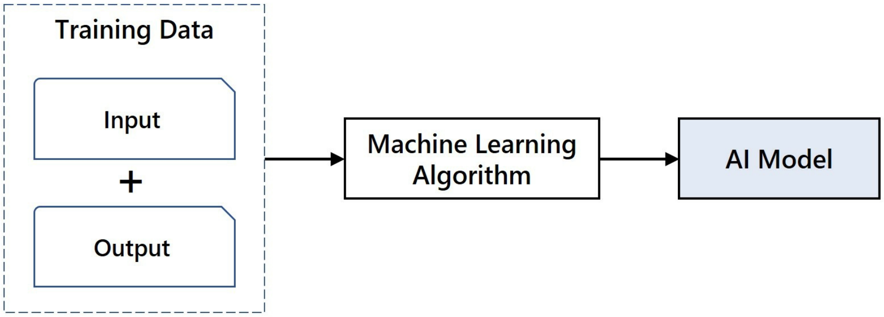


Figure 8: An overview of machine learning

The centerpiece of this learning process is ‘AI Model’. It encapsulates and assimilates the knowledge extracted by the learning algorithm. The best part is that it does so in an executable form. An AI model can be executed on a computer to predict or generate the output for new input data. Once trained, it works just like a conventional software program. It accepts input, and produces output. This ensures that a model can seamlessly integrate with conventional programs. This remarkable ‘ease of integration’ is the basis of comprehensive AI-powered systems. It lets multiple AI models combine in a sequence or in parallel to create powerful AI solutions.

From the diagram above, it may seem that an AI model is a program created entirely by the learning algorithm. However, that’s just half the truth. A model has two parts: an architectural part and a learnable part. The architectural part determines how the information is organized within a model. It defines the basic structure of the model and provides the placeholders for learnable variables of the model. It also defines the computation logic for the execution of the model. The architecture of a model is usually fixed at the design time, and is specified by an AI architect. The learnable part of a model is the intelligence captured by the model as per its architecture. This part comprises the values of the learnable variables. These values are used at the time of execution of the model, as per the pre-defined computation logic. The role of the learning algorithm is to fill in the learnable part.

For an illustration, let us consider a weighted decision matrix (WDM). It is a tool that we have been using for a long time. We often use it when we want to take a balanced decision while considering a number of influencing factors. It predates AI and even computers. Yet, it is an apt and simple illustration of how a model stores intelligence. Though a weighted decision matrix is usually a hand-configured model, it would work just the same if the weights were to be learned from training data.

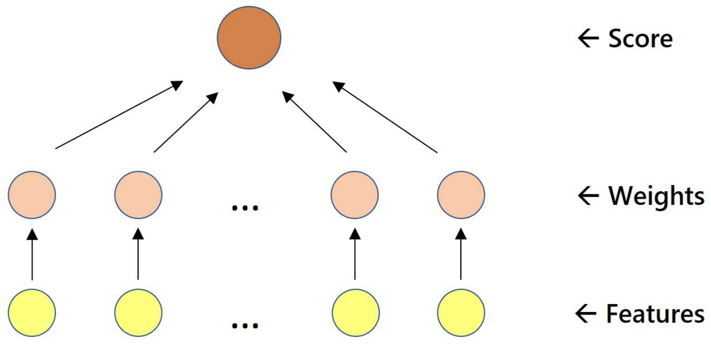


Figure 9: An overview of the weighted decision matrix

The structure of a weighted decision matrix comprises input features, weights and a score. Each feature represents an influencing factor that has to be considered in decision-making. All the features are known at the design time. One weight is associated with each feature. A feature's contribution to the score is computed by multiplying its value with the associated weight. The final score is computed by adding the contribution of each feature. The structure and the scoring logic complete the architectural part of WDM. The weights represent the learnable part of WDM. Of course, in a conventional weighted decision matrix, these weights are not learned from data; they are decided by the user. Higher the weight, more is the importance that is given to the associated feature. A weighted decision matrix is commonly used when one has to choose between multiple options. A comparison of the scores for these options guides the decision. This way, it captures the decision-making intelligence through its architecture and the values of the weights.

There are several classes of AI models, like neural networks, decision trees, Bayesian networks, etc. They all have an architectural and a learnable part. Take the case of a neural network: the number of neurons, connections, layers, and computation logic are all defined at the design time. They constitute the architectural part of the neural network. Values of weights and special variables constitute the learnable part. A machine learning algorithm works towards finding the best values for these learnable variables. For that, it processes the input-output pairs provided in the training data.

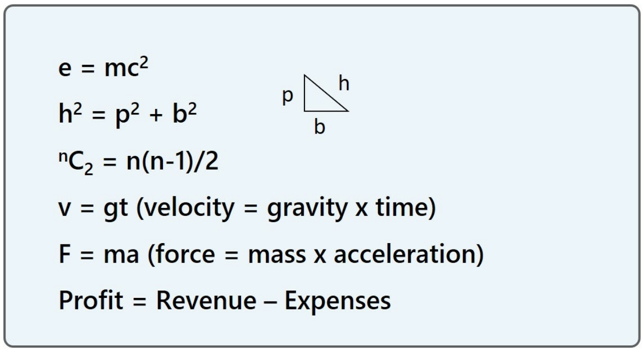
There is a symbiotic relation between a model’s architecture and the associated learning algorithm. They work in tandem. A model’s architecture defines the intelligence or knowledge it can represent. A learning algorithm learns the model from data as per the architecture. Its purpose is to fill in the learnable part of the model. A learning algorithm cannot learn anything that is beyond the scope of model’s architecture. Simultaneously, an architecture is useful only if the model can be learned from data through an algorithm. When we design new model architectures, we have to give due consideration to the learning algorithms as well.

The progress of AI depends on expressive power of AI models to represent different kinds of intelligence. A single architecture cannot capture all kind of intelligence, no matter how much training data we have. For example, a decision tree cannot capture the intelligence required to translate a sentence or identify an object in a picture. We need several kinds of model architectures to store the different kinds of human intelligence, skills and know-how. The AI fraternity is always looking for new and better models. AI engineers are constantly seeking ways to improve the capabilities of AI models. This is an area of intense research and development. In some domains the existing heuristics help to find the right models. In others, it is more of a trial-and-error exercise.

“The quest for AI is a quest for new kinds of model architectures.”

To design good models, we constantly seek inspiration from multiple sources. We seek motivation from each field that has helped us further our knowledge, skills and knowhow. We seek inspiration from multiple fields including math, logic, biology, psychology, human behavior and more. We seek ideas for both model architectures and the algorithms that can learn it. That’s how we push the envelope of AI.

Math-Inspired Models



$e = mc^2$

$h^2 = p^2 + b^2$

${}^{n}C_2 = n(n-1)/2$

$v = gt$ (velocity = gravity x time)

$F = ma$ (force = mass x acceleration)

Profit = Revenue – Expenses

Figure 10: A few famous equations

These are some of the famous equations from the world of science, math and economics. They encapsulate a vast amount of knowledge, in a succinct form. For example, the velocity of a falling object, after 3 seconds of free fall, can be quickly computed as 3 x 9.8 = 29.4 m/s. Or, the number of matches in a round robin tournament with 8 teams would be ${}^{8}C_2$ = (8 x 7)/2 = 28. Such computations may not be as predictable and accurate if this knowledge was expressed using plain text. The beauty of these equations is in their clarity and brevity. The equations reduce the scope for mis-interpretation. They provide a finality to the gathered knowledge, and set it up for deterministic use by everyone.

For centuries, we have been using mathematics to represent knowledge in diverse fields. Mathematical formulas and functions are proven tools that summarize the insights gained from years of research work. This influence of mathematics is equally applicable when it comes to learning from data. Statistics and probability are two fields that explicitly learn from data. For long, statistical analytics has been a guiding force for policy decisions and planning. Well observed probability numbers are the basis of decision-making in several fields. Even estimated probability numbers serve as good basis for decision making, especially when the decision depends largely on the underlying risk.

Given this backdrop, it is quite natural that we fall back on math-inspired ideas to design AI models. The models that are based on concepts from different branches of mathematics, such as logic, algebra, geometry, statistics, and probability. That’s how we use the likes of decision trees, support vector machines and Bayesian network as AI models. Most of these classes of models predate machine learning. They are well established to represent insights and knowledge across domains. At times, some of these models are directly used in conventional programs. The likes of hand-configured decision trees are routinely used to encode business knowledge.

The mathematical foundations of these model architectures inspire confidence in their capability to assimilate knowledge. The algorithms for learning these models from data, are well known for decades. With the availability of big data and increased computing power, it is now easier and more fruitful to run these algorithms. Many of these models and learning algorithms are rooted in statistical analysis of data. They are very effective at capturing dynamic intelligence.

Most math-inspired model classes have well-defined templates for the model architectures. The structures of decision trees and Bayesian networks act as templates. These templates guide the architecture definition through a few tunable parameters. The execution logic too is well defined by these templates, and only a few tweaks may be required. Thanks to math, the architecture, learning and execution of these models are well understood and quite predictable.

Brain-Inspired Models

When we are trying to build artificial intelligence, it is natural to be inspired by the working of our own brain. After all, replicating it is an aspirational goal of AI. Though physically it takes limited space in our head, its size is astronomical when we consider its building blocks: the neurons and synapses. Our brain has about 87 billion neurons. Each neuron is connected with a few others through links called synapses. These connections run into trillions. Each time our brain receives a stimulus, neurons in some part of the brain get activated. On receiving a stimulus from its connections, a neuron may sometimes fire to transmit it further to other connected neurons. This is how the neurons react to stimulus and communicate with one another. In simple terms, our thinking, knowledge, memory and thought process are believed to be the results of such communications between neurons.

We do not fully understand how our brain works. We do not know well about how to simulate brain’s neurons in a computing environment, even on a small scale. However, that is not necessary. The aim is not to create another brain, at least as of now. What we know about brain, is sufficient to inspire the design of neural network models. And, these neural network models have been very good at assimilating knowledge or intelligence. They are especially good for sensory intelligence. They have been very effective in the areas of language processing. They are the most promising models in the areas of design intelligence.

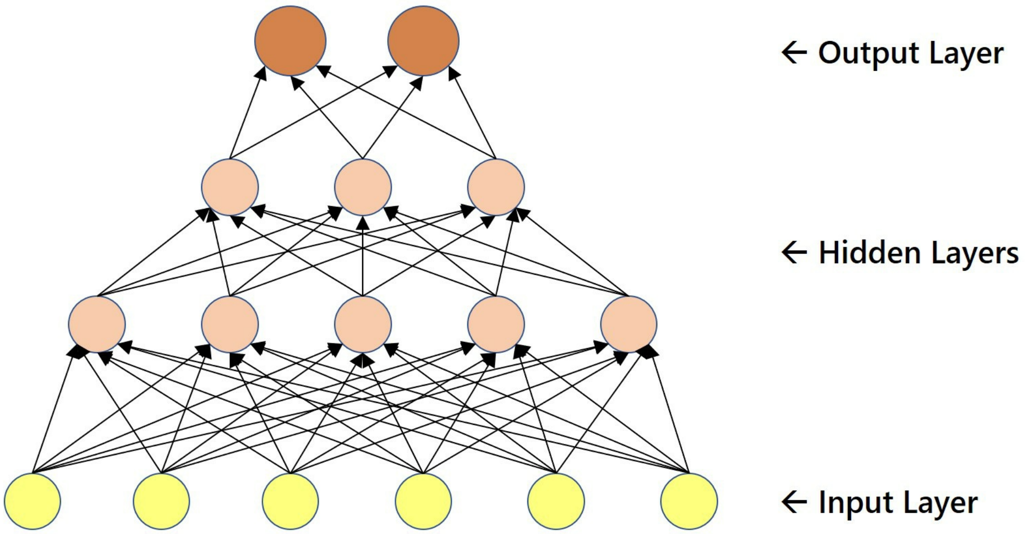


Figure 11: A simple neural network

Neural networks owe their exceptional power to two special properties. One is the support for myriad architectural styles, and the other is the scalability of the learnable part. The structure of a neural network is not constrained by mathematical concepts. Neural networks support free form architecture. The learnable variables can be organized in arbitrary arrangements of neurons, connections, layers and blocks. Special variables can be introduced in any block. The computation logic can range from simple multiplications to sophisticated mathematical formulas. We already have a number of architectural styles that are well established. There are likes of convolutional, recurrent and attention-based neural networks. Newer styles and templates are being designed all the time.

"The beauty of neural networks is in their super flexibility to support innumerable architectural styles."

This exceptional versatility in structure is enabled by a wonderful learning algorithm called backpropagation. The primary idea of this learning algorithm is quite simple. Its simplicity defies the complexity and scale of the math involved. A training sample is passed forward through the network. The computation logic is applied and the output is computed. If the output is not correct, then each learnable variables in the network is adjusted to reduce the error. This improves the chances of correct prediction with the same sample. If the output is as expected, then no adjustments are made, and the learning moves on to the next training sample.

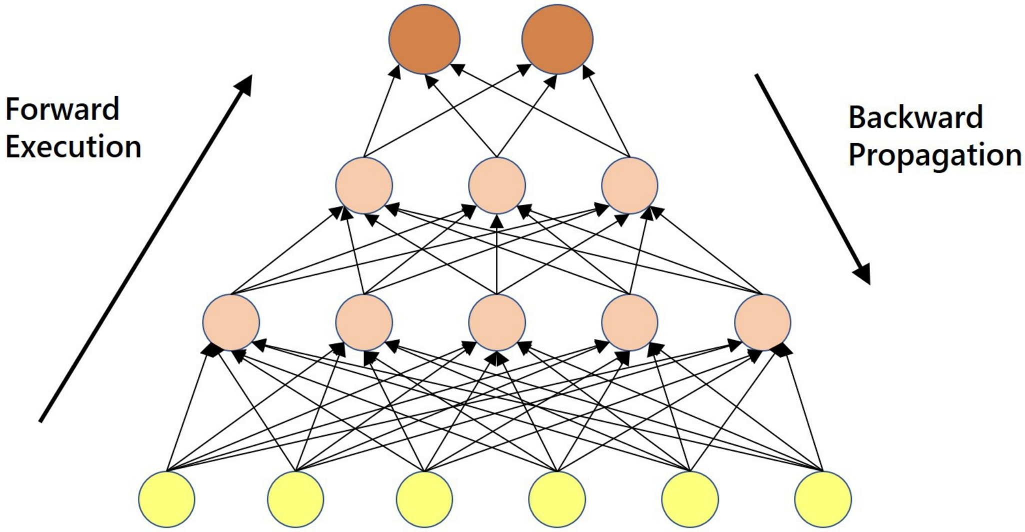


Figure 12: Learning through the backpropagation algorithm

The backpropagation algorithm propagates the error correction in backward direction. This is done by starting from outer most layer, up to the first hidden layer. The adjustments to the weights and other learnable variables, are computed by using advanced mathematics. To determine the adjustment-factor the algorithm uses the forward computation logic itself. That's almost like a magic wand. Through this, just about any architecture becomes learnable. Given this, the designers are free to focus on how the model would assimilate intelligence and execute it. The network designers can introduce special variables, define arbitrary computation, and repeat blocks of structural patterns. The backpropagation algorithm would still work in tandem with the architecture, and train the model.

“A neural network’s computation logic, which executes the trained model, is also instrumental in its training.”

A human brain has about 87 billion neurons. There are very few animals on earth with higher number of neurons in their brains. Some elephants and whales have over 200 billion, but most other animals have smaller brains. An ant has just about 250K, a honey bee has about 1m, a pigeon has 300m, and a giraffe has about 10b neurons in its brain. While a larger brain does not guarantee higher intelligence, it does help. It helps in information storage and processing. It helps with memory and skills. This benefit of scale is not limited to the animal kingdom; it helps in AI models as well. Scale is an important factor in the success of neural networks. It helps to try out different model sizes. The capability to assimilate knowledge often increases with the rising number of learnable variables.

GPT3 is a neural network model from OpenAI, an AI research and deployment company. GPT3 has about 175 billion parameters. The term GPT stands for Generative Pre-Trained Transformer. It is a language model trained for language processing tasks like translation, question answers, reading comprehension, inferences, etc. As described in the GPT3 paper, the team that trained this model also trained additional models, but with fewer parameters. They trained additional models with 125 million to 13 billion parameters. All of them adhered to the same architecture style. It is observed in the paper: as the model’s capacity increases from 125 million to 175 billion parameters, so does its performance on various tasks. The performance difference for some tasks was of many orders of magnitude.

“Scalability is a power source for neural networks. An architecture style can be scaled from a few hundreds to billions of learnable variables.”

GPT3 is an example of how the performance and capabilities of a model can increase with the rise in the number of learnable variables. The primary reason for this is that larger models can assimilate far more information, knowledge and intelligence. Of course, there is reasonable criticism of this idea. A very large model may actually be memorizing and not learning, simply because it has the capacity to do so. However, in general it helps to scale up an architectural style, definitely up to a point. Often, this is necessary because of the large number of input parameters, which we have to deal with. It is required in several domains including language processing.

The scalability usually comes at a price. The cost and time required to learn a model can increase rapidly as the number of parameters rises. The training data requirements can increase dramatically with rise in number of learnable parameters. The cost of training a super big model could be so high that only a few organizations may afford it. They can still be very useful in the areas where universality of intelligence rules. The benefits of 'Learn Once, Use Forever' may justify the investment despite the costs and risks involved.

In life, there are no free lunches; so is the case in the world of neural networks. With super flexibility and scalability comes along super chaos. When anything is possible, designing what works can be quite tough. Finding a fit-for-the-job architecture may become a matter of conjecture. Something that would require imagination, a lot of trial & error and then something more - good luck. However, all is not left to luck. There are templates, styles and heuristics available to design good neural network models. They can be directly reused or can guide the evolution of new architectural styles.

Seeking Inspiration

In conventional software engineering lifecycle, there is an analysis phase. It largely focuses on analyzing the domain and the requirements. Concepts and tools like object models and business process models are used to represent the domain. They are also used to capture and communicate the desired characteristics of the proposed system. The question is, can we learn object models and process models from some kind of domain data? Say through some interview questions or from the available descriptions of the existing systems? Can we learn them by observing how the current set up works? As of now it does not seem to be feasible. However, these are valuable models, and if they can be learned then it would mark a significant milestone in AI's progress. Object and business process models can be sources of inspiration to design new classes of AI models.

There are many more such areas of inspiration, which have the potential to help us build better AI models. As the work on AI progresses, we would figure out how to use them. We would see more of them powering the AI systems in future.

Summary

Models are the artificial brains with dual purpose. They store intelligence, and they also apply it.

Model architectures and learning algorithms work in tandem. Models provide the execution brain, and the algorithms provide the learning brain. They complement each other.

Neural networks are not bounded by mathematical concepts. They have the conceptual freedom to support numerous architectural styles.

Backpropagation algorithm is magical. It not only works with different architectural templates but also scales well to billions of learnable parameters.

4 DIGITIZATION: A FOUNDATION FOR AI

Numbers are the staple food that machine learning algorithms consume. A computer learns to recognize a face because the image is presented in the form of numbers. A virtual personal assistant recognizes spoken commands because it receives the voice in digitized form. A model that translates text from one language to another, sees the words as a long series of numbers. In fact, if there was no efficient numerical representation of words, language processing would not have been so successful. We might not have seen the language translators, which are so widely used.

It’s quite clear that machine learning thrives on numbers, so the natural question is - how do we convert images, sounds, words, and more into numerical forms? It happens through the process of digitization. Digitization creates a digital representation of things, whether tangible or intangible. It abstracts the world around us into a numerical form. It makes the world available to machine learning algorithms. Most learning algorithms work through the mathematical and logical processing of inputs. Thus, digitization is truly the foundation of AI.

The good thing is that digitization is a long-standing phenomenon. It predates machine learning, and in some cases even computers. Decades of computer usage mean that a large fraction of world’s information is already digitized. Thanks to the digitization of text, voice, and video, people today easily interact with one another through different devices. Money moves around the world in digital form without the involvement of paper of any kind. Crypto currencies exist primarily in the digital world. A large fraction of new content originates directly in the digital form. This includes documents, designs, and even the artwork.

Digitization is not limited to conventional computers and phones. Microprocessors are now embedded in several kinds of devices. Be it a still camera or a video one, the usage of films is long gone. The content is digital to begin with. A car has a number of sensors that continually monitor its performance and the health of its critical parts. The data from these sensors originates in digital form. That’s not all; a car also transmits this to designated servers for further processing. The data can be used for general improvements, assistance to customers, or for pure analytics.

The words digitization and digital are acquiring new meanings all the time. Banks nowadays are aspiring to be digital. It is not that they were slow to adopt computers or that they did not keep the data in digital form. The term is more about moving away from physical interactions with customers. To many, digitization means an opportunity to work remotely, whether it is office work or factory work. It won’t be a surprise if a true digital corporation comes to be defined as one where everyone works remotely.

The Enabler and the Barrier

When it comes to AI, digitization is like a double-edged sword. It is both an enabler and a barrier. Machines learn by processing numbers, and yet so often it is the digitization that makes the learning intractable. Historically, digitization was designed for storage and retrieval of the objects being digitized. It was not meant to aid machine learning. The historical designs do not care about the number of features that are created for an object being digitized. In the AI world, the phrase ‘curse of dimensionality' reflects the learning issues caused by large number of features. This phrase never appeared on the radar of these designers. Further, the digitization process was not designed to capture the semantics of the object being digitized. That was rarely of use for storage and retrieval, so never a design goal. For example, computers can easily save and retrieve text. However, the basic digital representation of a word does not reflect its meaning. A text’s digital representation is unsuitable for learning about language. Given this background, often the existing digitization is not suitable for learning.

Can we digitize the world, or rather the abstracted world, specifically towards machine learning? We surely can, and we have made significant progress. Still this field is in research & development mode, and we have a lot of work to do. In many domains, it is an uphill task that requires some breakthrough ideas. In a later section, we discuss one such idea: “digitization through relativity.” This idea has been successfully used for learning-friendly digitization.

Digitization itself creates multiple challenges that learning algorithms have to overcome. In the next few sections, we explore some of these challenges and their potential solutions. We begin with ‘Sensory AI’, an area of AI that faces these challenges the most.

Digitization Through Decomposition

Sensory intelligence is about the brain behind our senses. This is what gives meaning to what our senses observe. Like recognizing the objects that we see, differentiating between multiple kinds of sounds, and gauging the temperature of the objects we touch. We cannot fully explain how we really do it; certainly not in mathematical terms. Yet these are amongst the most prolific tasks we keep doing almost all the time. No wonder, we want machines to imitate and excel at what we apparently do so effortlessly.

In the world of machines, we have devices and sensors that mimic our senses. We have cameras for sight, microphones for hearing, sensors for certain kinds of touch and so on. Many of these devices go beyond our natural sensory capabilities. We have infrared cameras, ultrasonic devices, radars, and many more that dramatically enhance our ability to sense the environment around us. What these devices lack is the brain that can meaningfully process their observations. A camera cannot detect faces on its own. A microphone fitted into a personal assistant device does not recognize the words we speak. A sensor can measure temperature but would not know if it is hot or cold in a given context. Most of these devices just act as interfaces or input devices. The good thing is that they can easily connect with computers to transfer their findings. We now need AI models that can meaningfully process the data that these devices share. That’s what ‘Sensory AI’ is all about. It provides the brain power to process the sensory data. And the biggest challenge it faces has its origin in digitization.

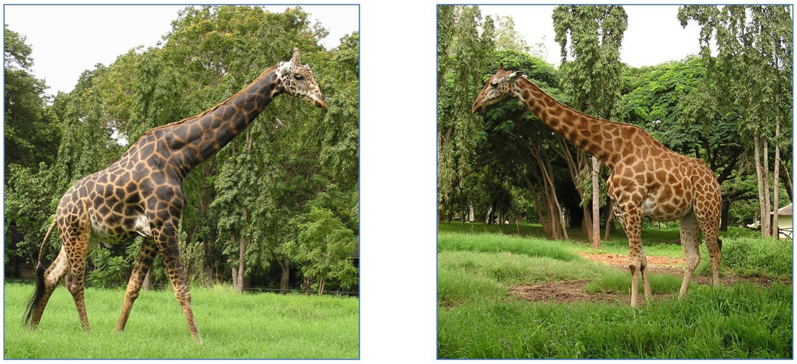


Figure 13: Identify the animal

Take a look at the two pictures above. Can you identify the animal in the pictures? It is quite likely that you would, provided you already knew about giraffes. In case you did not know, these are pictures of giraffes, an animal found in Africa. After knowing this, you are highly likely to recognize them when you see them in a zoo, jungle or even on TV. Just a few pictures of giraffe can be enough for any of us to recognize them in future. How does this work? How do we pick up the information needed to recognize an animal on the basis of just a few images? This is tough to explain, but one thing is for sure: we do not see these pictures as a collection of pixels. We do not learn about giraffes by processing a few million pixels of data. This is where humans excel at sensory intelligence. We do not need a digital representation of the world. We can make sense of it through some higher-order inputs than those provided by digitization.

"We humans use intelligence sans digitization. This is a huge advantage that we have over computers."

An image of size 100x100 pixels has 10000 pixels. In a monochrome image, each of these pixels can take any of the 256 values. A color image is lot more complex. A color pixel can take one of the possible 16.77 million values, or it gets split into three pixels reflecting the red, blue and green components. We cannot even imagine about taking decisions on the basis of 10000 features. If we were to recognize a giraffe by analyzing the numerical pixel data of an image, we would not succeed. Our brain cannot handle such large number of parameters simultaneously. It cannot take a rational decision on the basis of so many parameters. Back in 1956, psychologist George Miller came up with the ‘7 plus or minus 2’ rule. That’s the number of objects most people can hold in their working memory. The numbers 5 to 9 define our normal capacity to process information. In some domains, which we are highly familiar with, we may do slightly better. However, working with thousands of features is just not possible for us, even in familiar domains.

Representing an image through a set of pixels is an example of digitization by decomposition. In case of pictures, it is based on the spatial division of an image into pixels and selecting the dominant shade in that area as its value. Decomposition is used in other areas too. In the case of sound, digitization happens through sampling and quantization of analog signals. *The core idea is to divide the original object into smaller pieces and then record some measurable property of each piece.* With this method, the quality of object representation often depends on how fine-grained the decomposition is. For images, the higher the number of pixels, better would be the details of the image. For sound, a higher sampling rate leads to higher fidelity.

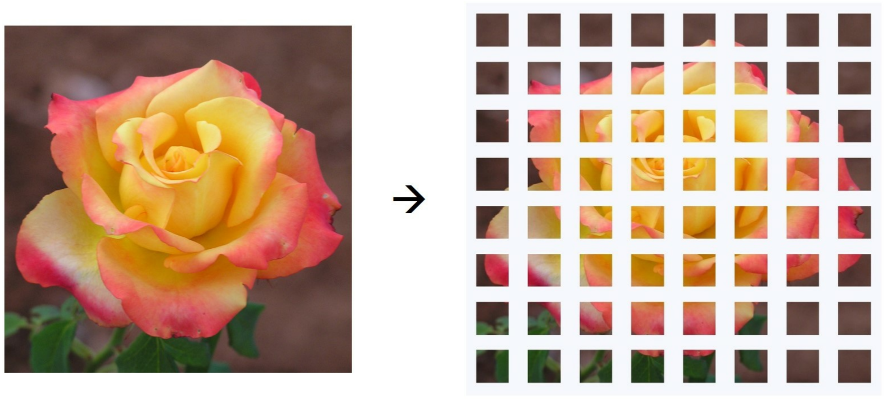


Figure 14: The basic idea of digitization through spatial divisions

This digitization by decomposition has major implications for machine learning. It leads to an explosion of features. Computers do have an advantage when dealing with large number of features. Unlike humans, they are not limited by George Miller’s law on working memory. They have huge memory and processing power to simultaneously process a large number of features. However, we still have to figure out how to use this processing power. After all it is not like – create a weighted decision matrix with ten thousand features, and it would recognize the object in an image. Since we ourselves do not need such digitization, we cannot directly help with the logic to process so many features. To extract intelligence from this digitized data, we need to find the right models and learning algorithms. Finding the right one is often a matter of chance. It can be a matter of trial and error. *That's why digitization that leads to a large number of features tends to be a barrier for AI.* Fortunately, most of the sensory intelligence is quite universal. Once we find the right architecture, the longevity of trained model compensates for the enormous effort.

"Computers can make sense of a large number of features, but only with the help of the right model architectures."

A picture of a giraffe can be taken from different angles, from different distances and in different light conditions. This means that there can be innumerable valid samples for a giraffe's image. Most of them may not seem too distinct to humans. Yet they could be distinct for machines that have to learn from digitized features. Each digitized image comprises tens of thousands of pixels (features), each of which can take hundreds of values. The potential number of distinct valid images is simply too high. It is not that a learning algorithm needs to go through all of the distinct images before it can learn to recognize giraffes. Machine learning is all about learning by generalization, and not through memorization. However, for a good model, the learning algorithms still need to go through a lot of distinct samples.

When the number of distinct valid instances is large, the learnable content from an individual sample tends to be miniscule. In a sense, we get weak signals of intelligence from an individual sample. This makes it tough to meaningfully learn from just a few samples. The solution to this problem is to aggregate these weak signals. This aggregation can be done by going through a large number of training samples. *The idea is to compensate for limited learnable content available with a single unit, by arranging for tens of thousands of them.* Digitization by decomposition is a major reason for these weak signals. It indeed creates a strong barrier, which requires a combination of good model architecture and big-data to overcome.

Large number of features affect not just the training data needs, they affect the model size as well. When an algorithm has to work with large number of features, it needs a model with huge capacity to store the learnings. The model must have high expressive power to assimilate the complex learnings. This is where neural networks excel. They have the versatility in model architectures, and they can easily support millions of learnable parameters. They work very well when learning with tens of thousands of features, by going through millions of samples. They are good at aggregating weak signals of intelligence from millions of training data units. Naturally, they are the preferred choice of models for sensory AI.

“Neural networks truly have the capacity to assimilate the sensory intelligence, and overcome the challenges presented by digitization.”

Digitization through Relativity

Computers have been working with text and language for decades. As mentioned earlier, text and documents are easy to store, retrieve, present and share. Text and thereby the natural languages are already available in a machine-readable form. However, this digital form is almost of no use when it comes to machine learning. The learning algorithms just cannot process this representation of text for learning purpose. The only processing, they can succeed at is comparison of one text with another. That’s too limiting for most of the Natural Language Processing (NLP) tasks.

Early attempts to use text for machine learning used a method called one-hot encoding. In this method, each possible word in a corpus has a designated position in a very large vector (a series of numerical elements). To represent a word, the vector element corresponding to the word is set to ‘1’, and rest of the elements are set to ‘0’. That’s how we get the term one-hot encoding. This encoding has no semantic meaning. It is almost like an identifier given to the objects manufactured in a factory. The identifier reflects none of the properties of the identified object. We cannot decipher any of the object’s attributes from this identifier. We cannot apply mathematical operations to this identifier as they would have no meaning. Though this one-hot encoding is useful for learning some kinds of tasks, it does not take us far.

The real aim of digitization of language should be to capture the intent of the spoken or written words. It would help when digitization assists the machines in deciphering the meanings of words and sentences. But is it possible? Can we digitize in a manner that lets machines make sense of words? The answer is yes and no. No, because so far, digitization does not reflect the absolute meaning of a word. It cannot capture what we mean by ‘nice’ or ‘strong’. However, it is possible to digitize in a manner that it captures the relative meaning of words. We can digitize words in a manner where machines can figure out that ‘positive’ and ‘negative’ are antonyms. Or that ‘rapid’ and ‘quick’ are synonyms. That’s what most of the popular word embeddings like Word2Vec and Glove do. These word embeddings are multi-dimensional vectors that represent a word. They are not one-hot vectors. Rather they are feature vectors with fixed number of features, each of which can take values that are real numbers. The relation between the meanings of two words can be computed by applying vector algebra to their feature vectors. We can compare the word embeddings of ‘good’ and ‘bad’, and the result would indicate that they have opposite meanings. Word embeddings as a form of digitization are not 100% accurate at capturing the relative meanings of words. Still, by using them, we can get exceptional results even for complex tasks like language translation.

Relativity is a great way to gain, organize and document knowledge. We know about the geographical location of a city by comparing it with carefully chosen but imaginary lines called equator and prime meridian. Most of us have mental map of places we frequently visit. Our home or city are the central reference point in this mental map. Performance of students in a class can be measured as relative to other students. Sports like tennis rank players relative to one another, and use this relative ranking to schedule matches in a tournament. Relativity does have some limitations, like we cannot compare students from different batches as the relative data about them is not available. Similarly, we cannot compare the tennis prowess of Roger Federer and Pete Sampras, or that of Steffi Graf and Serena Williams. They were all ranked as the No. 1 tennis player for over 250 weeks, but in different eras, and against a different set of opponents.

Word embeddings use the relativity principle to digitize words. It is not easy to define specific features that would represent the absolute meaning of a word. Word embeddings capture the relative meanings through feature vectors. They are mostly learned through statistical analysis of large corpora, which runs into millions of documents and terabytes of data. The accuracy or relevance of the word embeddings, tends to be higher for the domains, which were well represented in the corpora. This is an active area of research and newer methods of creating such embeddings are being designed all the time.

This method of digitizing through relativity has immense promise in other domains as well. It can be applied to many objects and situations where it is not easy to define specific numeric features. For example, if we have such feature vectors for food items, then it would help predict whether we would like a new item. All that we would require is to compare the feature vectors of a few food items. The vectors of the food items we like (or dislike) with the vector of the one we wish to predict for.

The vectors used in most of the word embeddings have between 100 and 2000 dimensions or elements. This means that learning algorithms have to work with very high number of features. No wonder, many of the natural language related models have to be learned from millions of data samples. They also require fairly large models to assimilate this knowledge. Significant amount of computing power is required to train and test these models. Word embeddings help a lot by digitizing the relative meanings of words, but learning through them still needs big data and large amount of computing resources.

Extending AI’s Reach through Digitization

When an organization is looking to hire new talent, ‘experience’ is one of the critical characteristics it looks for. Job advertisements routinely emphasize the experience that a candidate must have in a particular industry, technology or functional area. Companies are always looking for experienced managers, architects, engineers, testers and more. There is an underlying belief that several skills that are necessary for the job can be acquired only through experience. It stems from the fact that people develop certain skills only after having handled or gone through a variety of situations. Merely having studied about them is not enough.

Most experiential learning is constrained by time and space. It requires a lot of time to go through a variety of experiences. When we spend a lot of time in an area, we have higher chances of going through diverse experiences. There is also a major space perspective to experiences. We cannot learn everything about Java programming language by just developing web applications. For that we have to work on different kinds of applications and systems. This would normally require us to work with different organizations that are focused on different business domains. The story is same for skills in many disciplines, and not just Java language.

Can we speed it up? Can we gain the relevant skills without going through the time-consuming experiences? Can we gain diverse experiences without seeking diversity in our tasks and assignments? Special projects, workshops, trainings and going through the case studies would help. However, they help only up to a point. The effectiveness of substitutes for real experiences is limited.

Computers can transcend these limitations of space and time. They can go through just about any number of experiences and learn from them. They can do this at almost lightning speed, just like they learn other AI models. However, for that to happen, we have to first figure out what the machines should learn from these experiences. We have to figure out the kind of insights that have to be assimilated. The learnings we gain from experiences are neither standardized nor static. The learnings differ for different persons. We do not all learn the same things from same experiences, even if there is a broad overlap. Experiential learning is not in the realm of universal intelligence.

To make things worse, we do not know how to digitize the context for these experiential learning situations. We do not know how to represent these situations in a digital world. For example, as a football coach, we may keenly observe a match, and gain some strategic insights for the team to follow. However, we do not know how to represent the match in a digitized format for the computers to make similar observations. Similarly, we may observe a management case study, and learn from it. However, we are far from digitizing it into relevant features for computers to learn. When we succeed in this, a lot of experiential intelligence will be made available to the computers.

“The limits of AI are often due to the limits of digitization.”

When we can digitize both the context and the learnings (labels), computers may learn in minutes, what it takes years for us. Digitization is a critical piece in the progress of AI. Learning-friendly digitization of new classes of information would extend the reach of AI to new domains.

Summary

Machines extract intelligence from numbers, in form of numbers, and by using more numbers for doing so.

We have a major advantage over machines; we do not need digitization. Neither for learning nor for applying our skills.

Explosion of features often leads to weak signals of intelligence per unit of training data. To compensate, we must aggregate intelligence from a large number of training samples.

Just like models, ML friendly digitization too requires new ideas. Digitization through Relativity is one such successful and promising idea.

Computers can master many more tasks, provided we can efficiently digitize the context and the learnings.

5 AI-POWERED LIVE COVERAGE: A CASE STUDY

Sports Eye is a start-up that aims to take live telecast to every stadium, ground, court and field in the world. Its goal is to cover sports events live at every school, college, club and community center. It wishes to make live action available to all, from all kinds of sports venues. It plans to cover every event, whether professional or amateur. Thinking on conventional lines, these are ambitious goals, almost unachievable for the envisaged scale. Today, a professional video coverage of a game requires a large team of people. The team members have different roles like camera person, feed mixer, technical support, etc. There are multiple kinds of equipment involved. The work begins way before the event and continues well beyond it. Scaling up this set-up beyond a few venues is quite a challenge. The logistics and the availability of trained personnel are major limiting factors. Even if we ignore the costs for a while, we just cannot find enough trained people for the proposed scale.

"The business idea of Sports Eye is to capitalize on the huge upscaling opportunity available with the live coverage of sports events."

Sports Eye aims to use AI to overcome these limitations. It plans to build an AI system that would help it achieve the scale required for live telecasts. It intends to use relatively inexpensive and system-controllable cameras. These cameras would not require an associated camera person to operate them. The core idea is to build an AI-powered system that would churn out professional quality live coverage with almost zero human intervention. This system would have AI models that encapsulate the skills of camera persons, feed creators, replay selectors, and other technical persons. The system would know about the sport, and would quickly adjust to the venue. It would control the camera movements, prepare the live feed, and broadcast it. The following diagram presents an overview of the proposed system.

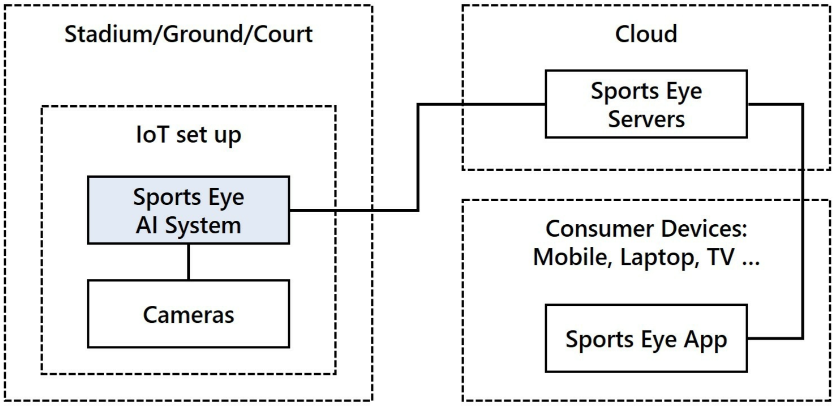


Figure 15: An overview of Sports Eye's AI system

In this chapter, we look at what it takes to build such a system. The focus will be on the AI part, depicted above as ‘Sports Eye AI System’. Specifically, we will pay attention to the AI models that are required for live coverage, and how they can be developed. The details that fall in the realm of conventional software will not be discussed. The topics of bandwidth, cloud, performance, availability, etc. are important but will not be covered in this chapter.

The idea of this chapter is to present various steps, heuristics, constraints, and challenges involved in building such large AI-powered systems. While Sports Eye intends to cover all kinds of sports, we restrict ourselves to the game of tennis. Occasionally, we will also refer to other sports like cricket and badminton, mainly to express the genericity of the proposed solution.

A reasonable knowledge of the game of tennis is assumed. Knowledge of service, point, game, set and match is assumed. It should be noted that a ‘game’ in tennis is a unit of score keeping. A set is won by winning at least 6 games. Most of the time the word ‘game’ would refer to the sports in general. Occasionally it would refer to the scoring unit in tennis. The context would make it clear.

As for the technicalities of video coverage, it would help to appreciate a few terms. A camera has pan, tilt and zoom movements. Pan refers to the left and right movements, tilt movements are along the vertical directions, and the camera can also zoom in and zoom out. A camera’s location is usually fixed. What it covers can be changed through its moves. A few cameras, like spider or drone cameras, do not have a fixed location, and can cover the game from multiple perspectives.

Covering a Tennis Match

Let’s consider a singles match in the sport of tennis. It has two players, one on each side of the court. The simplest way to cover this game is to use just one camera. It can be put well behind the base line at a height of about 10 to 15 feet above ground level. This camera’s coverage-frame is fixed, and there are no movements. All it does is to capture the video of the court irrespective of whether the ball is in action or not. Creating the live feed with such a set-up is quite simple. All that is required is to broadcast the captured video through the system. This does not require the system to use AI. Things fall entirely under the purview of conventional software systems.

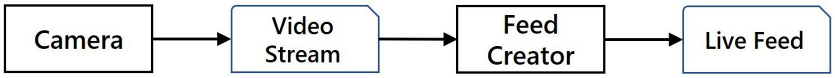


Figure 16: Basic coverage of tennis match with just one camera

While this is a reasonable setup that provides live coverage of the game, it is not an interesting one. We can certainly improve on it, even when we use just one camera. There are two avenues for that. One is to improve the feed without tweaking what the camera captures. The other is to improve what the camera captures by using the camera moves like pan, tilt and zoom.

In the first case, the live feed can be made more interesting by introducing replays of some of the points. This can easily be enabled by recording the live action and making the video accessible. However, recording and retrieving a video is the simple part. The real challenge is in taking the decisions regarding the replays, like – when can we show a replay? Can it be shown anytime or is there a specific time in the game to show it? It is quite obvious that we cannot show a replay while critical action is going on in the game. We cannot show it while a point is being played. It is best shown during the gap between the points or games. For this, the system must be aware of what’s going on in the game. It requires awareness of when the ball is in action and when there is a gap in the play. The system must recognize when a point has ended; only then can it introduce a replay. The system requires intelligence to maintain such real-time awareness of a tennis match. For that, it has to move towards the world of AI.

The second avenue to make feed interesting is to add new responsibilities to the single camera we use. We can use camera movements to cover the game better. It would reduce the monotony created by a fixed frame coverage. This too require awareness of the current state of the game. When the ball is in action, the single camera would normally remain in its conventional position. During the gap time, it can assume a secondary role. It can zoom in on players or show some spectators. This secondary role would require more AI. Intelligence is needed to figure out where a player is and how much to zoom in. The camera has to follow the movements of the player. The system would need to differentiate between players, officials and spectators. The system needs AI for such capabilities.

The awareness of what’s going on in the game, where the players are, and what they are doing, falls largely under the category of sensory intelligence. Thus, sensory AI would form the backbone of live coverage system. Besides this, the system should also possess knowledge of tennis, like the usual gap between two points at various stages of a tennis match. Or, which points are crucial in a match? A good coverage would combine basic tennis knowledge with the sensory awareness of the current state of the game.

To show meaningful replays the system would need skills that are even more complex. Like, recognizing the part of the action, which was interesting enough to show as replay. The replay-worthiness would normally depend on the importance of a point, or on the skills displayed during the point. The former is usually when a point decides a set, or when a crucial service is broken. The latter is about knowing what is a good display of skills. It is akin to recognizing beauty, which though lies in the eyes of beholder, is quite universal in the case of sports coverage. People do have a shared appreciation of display of good tennis skills. The challenges do not end here. We still have to decide where the replay begins and where it ends. This involves not just the aesthetics of the beginning and the end but also the available replay duration. After all a replay has to fit in the window provided by a gap in the action.

Today, the sports events are rarely covered by just one camera. Almost all kinds of games, be it tennis, badminton, cricket or football are covered by multiple cameras. Even amateur events are likely to be covered by more than one camera. Innovative techniques are being introduced every now and then. Games like tennis and cricket routinely use spider camera. Drone and robot cameras are also used to cover these games from multiple angles. This not only covers a game from additional perspectives, it also adds a feeling of being there at the venue. Multiple cameras certainly enhance viewers' experience. Of course, they also add complexity to the task of creating the live feed. There are more camera-angles to choose from, and more recordings to show the replays from. With multiple cameras, the AI system becomes larger and more sophisticated. It has more resources to maintain awareness of the game. Simultaneously it has more responsibilities to control and utilize these resources.

The Anatomy of Intelligence for Live Coverage

The intelligence required for live coverage of sports events is an interesting mix of universal and choice intelligence. The coverage is simultaneously rooted in repeating patterns and creative mix of camera angles. The former stems from the inherent characteristics of the game, and the latter from the opportunities provided by the availability of multiple cameras.

When we walk into a tennis stadium in the middle of a match, we can figure out what is going on. Maybe a point is being played, or the players are taking a break. Maybe one of them is about to serve, or there was a double fault. Maybe a service has hit the net, or it went out of the service area. Without even knowing the score, we can understand a lot of what is going on. This is possible due to a combination of our sensory intelligence and the knowledge of the game. The former gives us the power to observe, and the latter is instrumental in interpreting what we observe.

The picture of the entire match becomes clearer in our minds, once we know the current score. Now we know which player is leading, and by how much. We can make a better guess about how long the match may continue. We know how crucial the current point may be. We can appreciate whether a service break provided a critical lead or placed the ‘set’ back in balance. The score is a critical element for keeping a mental map of the game. The good thing is that this mental map is shared in the minds of most spectators. We all have a similar understanding of the current state of the game. We have almost the same appreciation of criticality of a point, game and set. The intelligence required to appreciate tennis is indeed universal.

In fact, this is true for other sports as well. The intelligence required to appreciate most sports is universal, be it badminton, cricket, hockey, volleyball or football. In each of these sports, we can appreciate certain aspects of a match without knowing the score. With knowledge of the score, we would have a more complete picture and a more involved viewing experience. Appreciation of a large part of any sports adheres to the concept of universality of intelligence. It is this universality that lets us communicate and discuss about a match, both during and after the match.

When we go to a stadium to watch a tennis match, we view the play from a particular seat. We cannot normally change our place to get a different perspective. Yes, we can move around a bit but that has its limitations. In sports like tennis, movements are not allowed when the play is on. We have to watch the game from single perspective. A live coverage of a game is not limited by this constraint. First, there are multiple cameras placed in some of the best coverage locations around the court. Second, the telecast can switch between camera angles within a fraction of a second. When we can choose any camera angle to show a game, it opens the doors for creativity. Indeed, the live feed is a creative mix of changing camera angles and well-chosen replays. In tennis the coverage is quite dynamic. They may use one camera angle just before the service, another during the play, a third one after the point, and show the replays from a fourth one. However, there is no rule that dictates this sequence. The camera angles can change, the durations they are used for can vary, and a few transitions may even be dropped. The telecast uses a few patterns with mini variations.

The use of multiple cameras to cover a game falls under the purview of choice intelligence. Yes, there are some rules and conventions to be followed, but the choices are plenty. There are good choices and standard choices, but no completely right or wrong choices. Breaking monotony is as much an objective as selecting the best angle to cover the game.

AI Models for the Live Coverage

Games like tennis, cricket and badminton are played as a series of a few repeating loops. In tennis, players walk on the court and take positions to play. One of them receives the balls to serve. There is usually a small routine of tossing the ball on the court before the player serves. Once the service is valid, the point moves shot by shot. When there is an invalid shot, the point gets completed, and the players move around a bit before taking their positions again. When a ‘game’ gets over, the service-end changes or the players take a break. This sequence repeats with minor variations, game after game and set after set, until the match concludes. There are a few deviations like a tie breaker, but even then, the flow of the play is not very different. The following diagram captures a high-level view of the repeating play-loop in a tennis match:

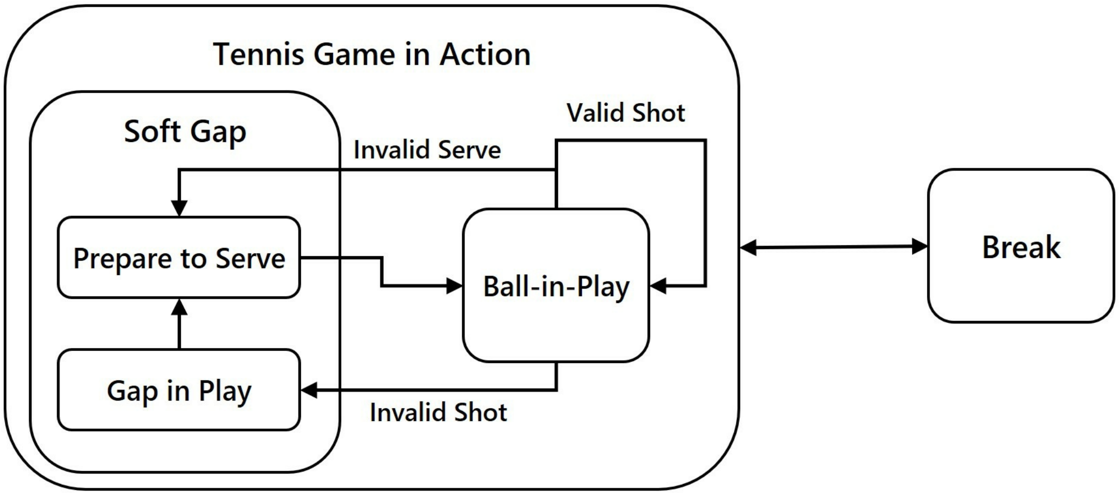


Figure 17: The play-loop for a tennis match

The play-loop is quite similar in the game of badminton. There are a few differences, like in badminton there is no second serve. The game of cricket has a conceptually similar flow. It has multiple types of breaks during a match. This is depicted in the diagram below.

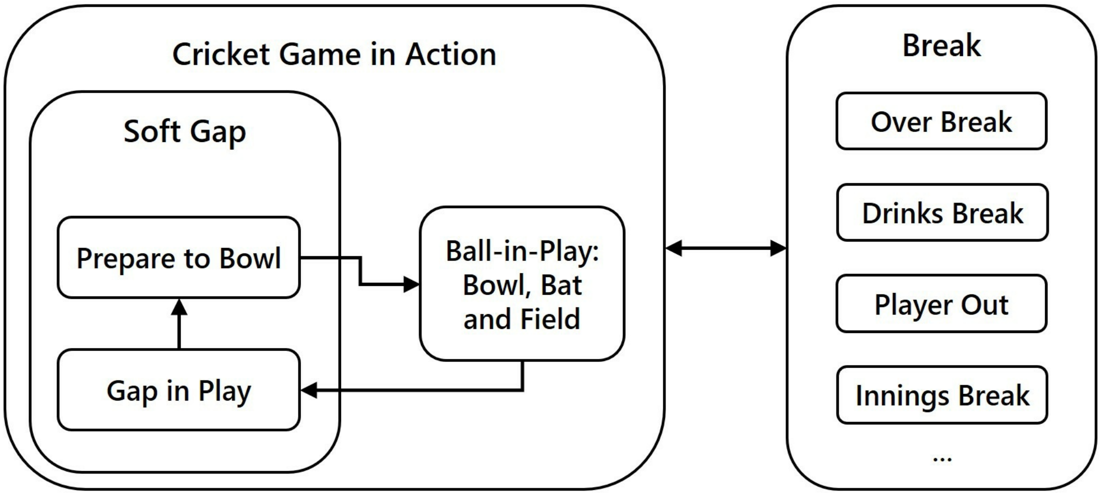


Figure 18: The play-loop for a cricket match

Each of these states and sub-states of the game is covered by using a few coverage patterns. There are subtle coverage rules with a few options for camera angles and content. In general, the breaks are used for advertising or replays. Soft gaps are used for replays and multiple views of preparatory actions. The intense action is covered largely through standardized routines and select camera angles. These coverage patterns repeat throughout the match.

The team that creates the live feed of a game has detailed knowledge of this play-loop. They are always aware of the current state of the game and the events that are happening in it. This awareness tells the camera persons about where to focus. It guides the feed creator to mix the video streams from different cameras. Needless to say, the AI system for live coverage would also have this knowledge and awareness.

The diagram below depicts the AI models that would form the core of the live coverage system. These models would interoperate to provide the live coverage of a tennis match. Some (but not all) of the inputs and outputs of these models are also presented.

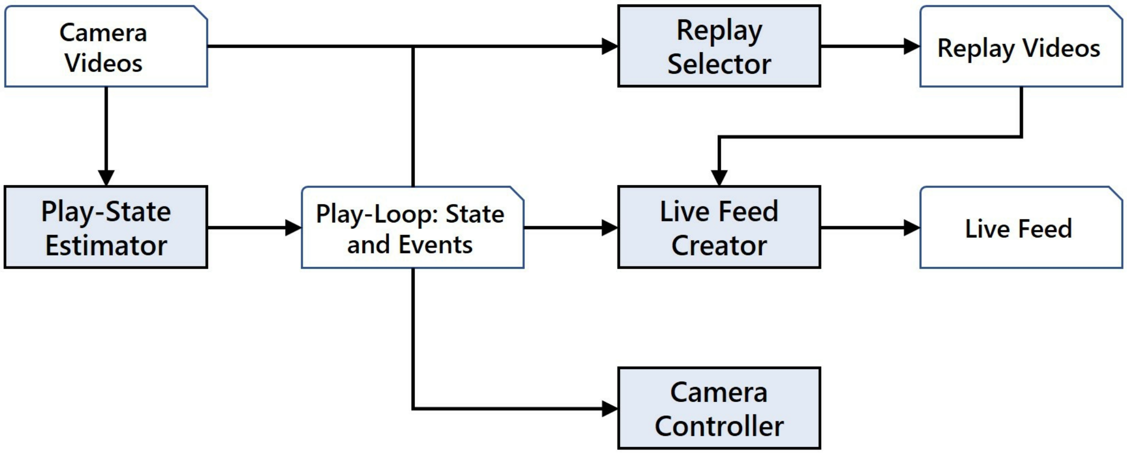


Figure 19: The AI models for live coverage

The model ‘Play-State Estimator’ has the critical responsibility of estimating the current state of the game. It would observe the activities happening on the court, detect specific events, and estimate the current state of the game. It would continually monitor the court for state transitions. This information would then be used by other models, which would control camera movement and create the live feed.

To estimate the current state of the game, ‘Play-State Estimator’ has to process the live visuals of the tennis court. These visuals are available from the cameras deployed to cover the game. The processing of these video streams has to be done in real time because the state of the game can change every few seconds. A good coverage is expected to respond to important events within a fraction of a second. This happens all the time in professional quality coverage. For example, a player focused shot is shown at the time of service, and the feed switches to the main camera before the ball hits the ground. Thus, the simultaneous and rapid processing of multiple videos is an important capability that has to be built into the AI system.

In most professional coverages of a game, the cameras shoot the game from multiple perspectives. These perspectives are defined by a few factors, like:

1. Frame: A camera may cover the game through a long, medium or close up shot. For example, a camera may focus on a player’s face, show a full picture of a player, show the player from a few feet, or

cover the entire court in a frame.

2. Tracking: A camera may continually change its direction to follow the ball or a player’s moves.
3. Location: The location is fixed for most cameras, and it is an important determinant of what a camera can capture. Further, a few cameras can change their locations to cover the game from different angles.

A camera may play multiple roles during the coverage of a match. It may be used to show a close up of the player who is preparing to serve, or a medium shot at the time of service. It may also be used to track the player across the court, both during a point and during the soft gap between points. A camera’s role decides its framing and tracking objectives. As there is no person to operate these cameras, these camera roles would be supported by AI models. These camera controller AI models would usually have three constituents. The first one would continuously detect a player or the ball. The second would predict the movements of the player or the ball. The third would adjust camera pan, tilt and zoom to continually meet the role’s tracking and framing objectives. This constituent would have higher complexity for cameras that can change location.

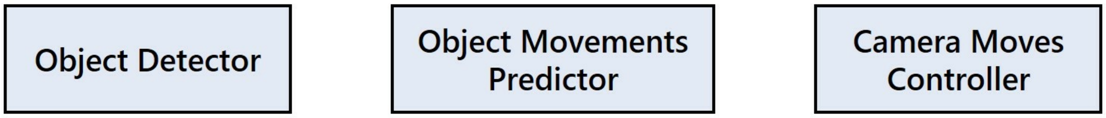


Figure 20: Constituents of a camera controller model

Live feed creator is an AI model that largely reflects choice intelligence. For almost all the states and sub-states of the game, it is possible to use more than one camera angle. While there are subtle rules and conventions to follow, the choices to create the live feed are plenty. They keep increasing with increase in number of cameras. Avoiding monotony is an important objective of the live coverage. This is achieved by choosing different camera angles and roles to cover the game. While most points may be shown from the primary camera, a ground level camera may be used to show some of them. One service may be shown with a close up, and another with a medium shot. Choices for replays too increase with multiple cameras, and they can be played at regular speed or in slow-motion.

With so many options to choose from, a live feed creator essentially comprises two policy constituents. One encapsulates the state-specific conventions about the coverage, and the other is a moderation policy. The 'conventions' part would be learned from the training videos. The moderator part would alter this policy as per the needs of a venue and a tournament. For example, in some tournaments we may take more advertisement breaks than others. In some cases, we may show spectators more often, and for longer duration. The live feed creator would be complex model that would balance the conventions with moderation policies.

State Specific
Coverage Conventions
Moderation Policy

Figure 21: Policy inputs to Live Feed Creator

As discussed in an earlier section, replays are a great tool to make the live feed interesting and informative. The task of replay selection is quite complex. The system has to decide which action is worthy of replay. It has to choose from the available angles to show it. At times a point or a shot may be shown from two to three different perspectives. While the timing aspects of replays are decided by the 'Live Feed Creator' model, the content part is supported by a 'Replay Selector' model. This model continually analyses multiple video streams of live action. It analyses them for being worthy of a replay. It stitches together replays of various durations, and also decides if one is to be shown in slow motion.

Model Training with Enriched Features

Video streams are the primary source of training data for Sports Eye's models. A video usually comprises 30 frames or images for one second of video. Each HD video frame has over 2 million pixels, and each color pixel can take one of the 16.8 million values. For learning purpose, each pixel of a frame becomes an input feature. Given the number and complexity of features in a video, it is quite a task to learn a model from video streams. Of course, the frames can be resized for learning purpose, but the number of features would still be too high. To learn a complex model like 'Play-State Estimator' directly from video stream, is an enormous task. The cost of doing so would be extremely high, and success may still be elusive. The complexity of the engineering aspects of learning process may also be intractable. Clearly, learning these models directly from video stream data is not a good option for Sports Eye. The models like 'Play-State Estimator' and 'Replay Selector' should be learned through alternative methods.

The conventional idea of machine learning is to learn a model from scratch. The general impression is that unlike humans, machines cannot benefit from prior learnings. Many books, talks and articles present AI in a manner that strengthens this notion. It seems that learning algorithms are not capable of benefitting from prior learnings. They cannot leverage the intelligence encapsulated in other AI models, to create a next level model. However, this impression is not correct. It is quite possible to leverage prior learnings for new ones. Given the enormity of the learning tasks that Sports Eye needs to accomplish, leveraging prior learning is clearly an alternative to work on.

When we watch a tennis match, how do we know about the current state of play? As discussed in an earlier section, we look for where a player is, his or her movements on the court, and what his or her pose is. We also track the ball movements across the court. We then derive the current state of the game from these elementary details of the current activity on the court. Coming to the AI system, this knowledge can be provided by elementary AI models like player tracker, pose estimator and ball tracker. We can use them to build the next level of models.

One of the ways to do so is to use enriched features as input features for the higher-level model. *Enriched features contain far more information than the raw features they are derived from. They abstract out critical information provided by thousands of features into just a few. This way, they provide stronger signals of intelligence per unit of training data.* For example, a player tracker AI model would tell us whether a player is in the frame or not. When the player is in the frame, it would also tell the size and position of the player with reference to the frame. This way the information about a player, which would otherwise require going through a large number of pixels, is abstracted into less than 10 features. Of course, we lose a fair amount of information in the process. Like we cannot use these 10 enriched features to predict the player's pose. However, that does not matter. After all, most intelligence depends on the abstraction of the real world from certain perspectives. It is the utility of the abstraction that we are concerned about, and not what has been left out. What we lose out due to one enriched feature, can very well be provided by another. For this, we can apply multiple elementary AI models to the same video clip. This way, we can extract all the useful abstractions while dramatically reducing the number of input features for training.

Enriched features are not new to us. We routinely use enriched features that are abstracted out from vast amount of underlying data. When a company goes for campus recruitment, it often uses a student's grades in the relevant subjects for the initial shortlisting exercise. A student's grade is an enriched feature usually derived from a number of tests, assignments and project scores. By referring to the grades the company avoids going into detailed performance records. Similarly, a credit score is an enriched feature that represents credit behavior on the basis of hundreds of data points. Its usage makes it easy to take lending decisions. Stock market indices like Dow30, Nikkei and Nifty50 are like enriched features. We can gauge general market movement by observing the changes in these indices. We do not have to monitor individual stocks that contribute to these indices. Creating or deriving enriched features does require processing a large amount of underlying data. However, it is worth the effort as *it makes the next level tasks easier.* When we use elementary models like player tracker and pose estimator, we are using AI to create enriched features. This way we use prior learnings to learn the next level AI model. The machines benefit from the AI problems solved earlier, and build on the expertise gained.

Before we go further with enriched features, we must explore a critical question. When is it possible to learn a next level model from prior learnings? The answer is – *when the intelligence required for a task can be defined in terms of that for relatively elementary tasks.* It is possible when we can decompose the required intelligence into its relatively elementary constituents. This is where the domain knowledge becomes critical. Can we know the state of the game just by keeping track of ball movements? Yes, we can find the periods of soft gaps and intense plays by keeping track of ball movements. Within soft gaps we would know that a player is about to serve by observing his or her position on the court. It is this kind of domain knowledge that helps us compose higher-level models in terms of more elementary ones.

For a model to use enriched features at execution time, it should have been trained with them in the first place. This means that an elementary model like pose estimator is first applied to the video stream, and its output is recorded. This output is used as input features for training the next level model like ‘Play-State Estimator’. The same process is repeated at the time of execution of the higher-level model. First, the data is fed into the elementary models, and then the enriched output is fed into the higher-level model. The following diagram captures the entire training process that uses pre-trained models:

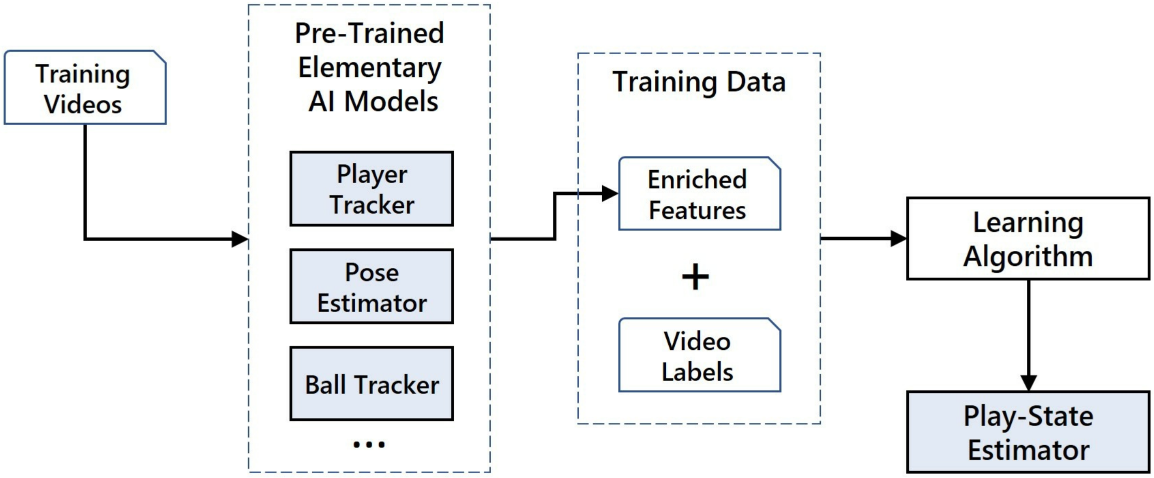


Figure 22: Model training with enriched features

Using enriched feature has a significant benefit with reference to the amount of training data required for learning a good model. *In fact, this benefit is so huge that it is a game changer for Sport Eye.* The higher-level model can now be learned from fewer features of lower complexity. This reduces the amount of training data required for the higher-level model. We need much fewer number of labelled training videos. This lowers the labeling costs for Sports Eye. *Here it shall be noted that we are not learning the elementary model from the same data or the same amount of data as the higher-level one.* At the time of training of the higher-level model, the elementary models are already pre-trained.

“Enriched features provide stronger signals of intelligence per unit of training data, and thereby reduce the training data needs.”

How do we create the elementary AI models like ‘Player Tracker’? *Here, we benefit from the universal nature of the intelligence required for these elementary tasks*. Sports Eye can buy best in class models for ball detection, ball tracking, player detection, player tracking, pose estimation and more. However, there may be an issue. At times the pre-trained model may not be well-suited for the game of tennis. There may be data coverage issues. The model may not have seen enough of the tennis videos, and would underperform. Sports Eye would do well to get pre-trained models that are suitable for tennis. In case that is not possible, it would still be good to get a pre-trained model and retrain it for tennis. This fine-tuning of a pre-trained model would usually require much less training data than required for one that is built from scratch. It will also have the benefit of fit-for-job model architecture. Further, its pre-configured learnable variables could be easier to adjust towards tennis specific usage. The pre-trained models, whether used directly or after fine tuning, would save a lot of time and effort. They would let Sports Eye focus on relatively more complex models.

Preparing the Training Data

A trained AI model is a reflection of its training data. Thus, the importance of training data cannot be overemphasized. Sports Eye aims to provide best in class viewing experience. Naturally, it would like to use the highest quality videos of the game to learn from. When it comes to tennis, the Grand Slam tournaments represent the best in terms of quality and arena of play. These tournaments also have best-in-class video coverage. For this case study, we assume that Sports Eye has access to the videos of a large number of tennis matches from the Grand Slam and other premier tournaments. It has thousands of hours of videos. It not only has the live feed videos but also the camera specific videos, only a fraction of which make it to the live feed. These additional videos are important to learn about the roles a camera performs at different times and how it does them. We assume that these camera specific video streams are available with timestamps that are in sync with live feed. This would help in figuring out how the action on court appears from different perspectives. It would enhance the learnability from multiple video streams. It can be safely assumed that this video data contains all kinds of situations that can occur on a court. We also assume that the quantity is good enough to learn most of the models discussed in the previous section.

It is necessary to have a large quantity of raw video data, but that is not sufficient for learning. We require labelled data to learn from. Raw data would become training data only after it has been appropriately labelled for the concerned models. Herein lies the challenge for Sport Eye. In fact, a huge challenge. Labeling a video is a complex exercise, even when someone knows what and how to label. We have to mark the point at which a particular event happens or an activity begins. For states and sub-states of the play, we also need to mark the end points. This is a time-consuming process, and it is prone to labeling inconsistencies. Managing it would require special tools for labeling. Sports Eye will have to develop or buy tools that can manage large scale labeling activity. It would require a complete training data management tool. This tool would keep track of multiple related video streams, segments and labels. It would allow a video stream to be broken into multiple overlapping segments.

During the labeling exercise, an expert would provide labels to different video segments. These labels would reflect a state or a sub-state of the game. They may also reflect events like a service or the end of a point. The entire labeling exercise can be expedited if the system can detect potentially interesting segments. The user would then have to just label these pre-marked segments. As and when required a user may split a pre-marked segment or merge a few of them. A user can also create new segments, which may overlap with existing ones.

One of the ways to automatically mark segments is to use the live feed. A live feed comprises multiple video segments picked up from different cameras. These segments are easy to recognize through software. The boundaries of these auto-recognized segments can be projected onto time-synchronized videos from different cameras. This way we can create reasonably meaningful segments for all the camera streams. This works mainly because the live feed segments tend to reflect the state of the game and important events during the game. These segments would make it easy to mark labels like ‘break’, ‘soft gap’ and ‘ball-in-play’.

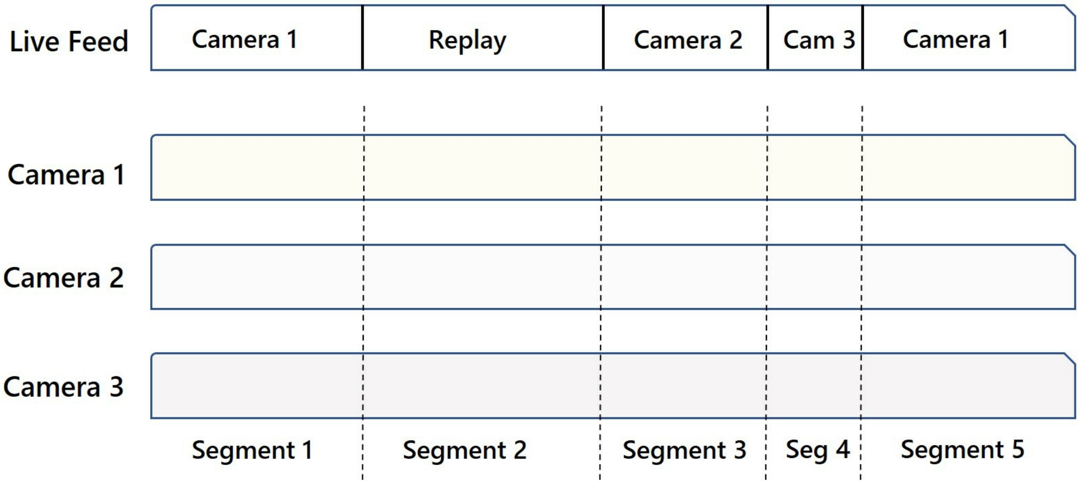


Figure 23: Creating video segments through live feed

The labeling of data has to be in sync with the requirements of the models to be learned. As we have multiple models to learn, a video segment may have to be labeled from multiple perspectives. There can be one kind of label for recognizing the state of the game and another for replay selection. The labeling tool should have the facility to record multiple labels for each segment. This way a video segment can be used to train different high-level models.

As discussed in the previous section, most of the high-level models like ‘Play-State Estimator’ and ‘Replay Selector’ would be trained with enriched features. For this, the video clip of a segment should be passed through the elementary AI models. The output of these models should be processed to convert them into enriched features. These enriched features can then be recorded with the concerned segment. The training data management tool should facilitate this. *The availability of segment-level enriched features along with the labels would help expedite the model development work.*

Moving on to the Amateur World

"A trained model should be used on data that is similar to its training data". This is a primary principle of machine learning. The training data should reflect the data that the model will be used on. It is true that good models generalize and not memorize, but this generalization has limits. *By and large, the trained Models work well on data that can be interpolated from training data samples.* They also work reasonably well with a small degree of extrapolations. So, generalization works within training data limits, and not much outside it. It is tough to guess the scope of this generalization, especially when we work with a large number of features. What appears to be similar data to humans may not be so for the model. We have to be prepared for surprises and be ready to take corrective action.

When we use the training videos from Grand Slam tournaments, will the models work in amateur situations? How successful would the system be on courts that do not match the grand slam standards? Would it work on courts in schools, universities, and clubs? These courts may not even have good places to fix multiple cameras. The camera angles used in training videos may not be feasible. There would be fewer cameras, which may make it tougher to estimate the state of the game. In amateur situations, the players would not take standard breaks, and they may not change service ends. They may not wear attire similar to that worn by professional players. In many of these places, there will not be any ball boys or ball girls. The umpire would not sit near the net. Even the score may not be displayed anywhere.

How would the system cope with these differences? Would the trained models and the overall system work well? They may not. We have to take cognizance of this possibility and design the system to deal with these variations. We have to build safeguards right at the design stage. Finally, we have to be prepared to adapt the system through new and updated models, as and when required.

There are quite a few design elements in the Sports Eye's system, which make it capable of handling diverse real-life situations. While it may seem that Sports Eye’s AI models are built solely on training data from the premier tournaments, it is not true. The system has much wider coverage of real-life situations. The system uses a large number of pre-trained model for elementary AI tasks. *These elementary models are picked from the market and would normally be trained for diverse conditions.* Their training data would include videos way beyond the premier tournaments. The higher-level models of Sports Eye are largely trained with the enriched features that are provided by these elementary models. This gives the overall system an ability to operate well in diverse situations.

The camera controllers too use these pre-trained elementary models. The elementary models are used for detection and motion prediction parts. This makes camera controller learnings fairly independent of the angle from which they cover the court. They are likely to work well in non-standard camera positions, which may be the norm in amateur venues. These camera controller models add robustness to Sports Eye’s AI system.

The Live Feed Creator is one model that needs special attention. It has to be designed to withstand the errors in estimating the current state of the game. It has two important constituents. The first one statistically learns about camera choices that can be used during different play states. The second one implements a configured policy to moderate these choices. This is a programmed component. *It can be programmed to make some default choices when other models are not able to provide adequate information.* It can provide feed from the default camera when a more informed decision cannot be taken. In the same manner, whenever required the camera controller too can instruct cameras to switch to some default coverage modes. This would usually be a fixed and wide coverage of court. These default modes would help the system quickly regain a finer awareness of the state of the game.

The aim of Sports Eye is to provide the best coverage in all conditions. For this it should continually analyze new videos received from the deployed systems. These videos should be used to fine tune the models. Whenever required, new model variants should be trained for new and often-repeating conditions. Improving AI models is a continuous exercise, and Sports Eye should utilize all its field data to improve the viewer’s experience.

Summary

Sports Eye is looking at a massive upscaling opportunity in the live coverage area. This opportunity is enabled by a technology combo of AI, IoT and cloud computing.

The intelligence required to appreciate most of the sports is universal. This is a significant leverage point for Sports Eye’s business.

The high-level AI models can be learned through enriched features that are derived from pre-trained elementary models. This is a critical design pattern for the techno-commercial success of this system.

The usage of off-the-shelf pre-trained models lets the system perform in diverse amateur situations, way beyond the scope of its own training data.

Prudent usage of choice intelligence can help the system overcome the limitations of core AI models. This design pattern also paves the way for seamless upgrade of the AI models.

6 DOMAIN-DRIVEN AI ENGINEERING

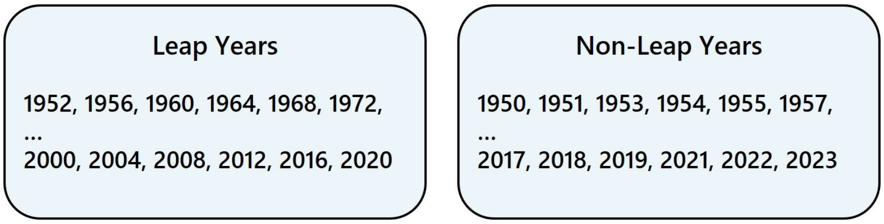
In the 1996 movie ‘The Rock’, the two protagonists, played by Sean Connery and Nicolas Cage, were chosen for a mission owing to their special skills. They had the critical domain knowledge necessary for the mission. As the mission progressed, their domain knowledge indeed proved crucial. Of course, it was a scripted story, yet it serves well to underline the importance of domain knowledge. Be it for a rescue operation or the development of a software system. Domain knowledge is often the secret ingredient that leads to highly successful software systems. Success rates are often lower when the software teams lack domain knowledge.

When it comes to building an AI-powered system, the domain knowledge is no less important. The task of learning from data depends heavily on the characteristics of the domain’s data. The role of domain knowledge starts with the abstraction of real-life objects and situations. It guides the ways features are selected, pre-processed and transformed. Domain knowledge is instrumental in the fine-tuning of learning algorithms and the preparation of training data. It has a profound influence on the architecture of AI models.

In this chapter, we discuss how domain knowledge is critical for training of good AI models. We also explore a few design heuristics that leverage it.

Data Coverage

Let’s consider the task of deciding whether a particular year is a leap year or not. We want machines to learn about it from training data, and then make predictions for the test data. Let’s assume that we have selected a good model architecture. We also have an appropriate learning algorithm, which can learn the logic to classify a year as a leap or non-leap year. We now provide the following training data. It comprises year numbers from 1950 till 2023 that are labelled as leap or non-leap years.



Leap Years

1952, 1956, 1960, 1964, 1968, 1972,
...
2000, 2004, 2008, 2012, 2016, 2020

Non-Leap Years

1950, 1951, 1953, 1954, 1955, 1957,
...
2017, 2018, 2019, 2021, 2022, 2023

Figure 24: Training data for the leap year model

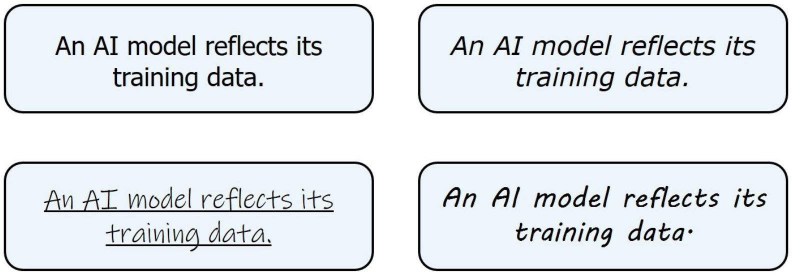
The algorithm learns from this data, and it creates a model that is ready to make predictions. We give this model 5 different years to make predictions: 1897, 1899, 1900, 1904, 1905. One of these is a leap year, and the rest are non-leap. What is the probability that the model will predict the year 1900 as a leap year? It is quite high, almost near 1. A well-learned model is likely to get all the predictions correct except for the year 1900. Why? From the data above, a good algorithm is likely to learn that a year is a leap year when it is divisible by 4. However, there is no data element, which tells the learning algorithm that a century year has to be divisible by 400, else it is not a leap year. The algorithm just cannot learn this rule. The trained model would divide the year 1900 by 4, and predict it to be a leap year.

This is a contrived example for machine learning. After all a leap year is determined by the rules defined by humans. We can easily write a conventional program with just 2 rules, and it would correctly predict leap years. Such a program will have just a few lines of code, and will work without any errors. Why make machines learn this from data? The answer is that we should not. However, this contrived example aptly highlights a crucial aspect of machine learning – data coverage.

Data coverage is an important factor for learning good models. While a model’s potential to assimilate intelligence depends on its architecture, the actual learning is determined by the training data. We know that a conventional program would be deficient if the requirements specifications do not contain relevant rules, checks and validations. Let’s say the business analyst for the conventional program forgot to specify the ‘divisible by 400’ rule for century years. In such a case, the conventional program would err for some of the century years. Similarly, an AI model would be deficient if its training data did not cover all the critical variants present in the domain. In a sense, *what gets specified as requirements for conventional programs, has to be provided through good data coverage for AI models.* The role of a business analyst or requirements engineer thus gets extended for AI initiatives. For conventional programs they have to study the domain and business processes to specify requirements. For AI initiatives, they also have to analyze training data and ensure that it covers all relevant variants that would map to the intended labels.

"An AI business analyst must identify and include all significant data variants into the training data."

However, this can be a cryptic task. It requires both domain and AI knowledge to accomplish. The analysts and architects must study the domain from machine learning perspective. What is a significant data variation for machines may not be so for humans. Let's have a look at the following text:



An AI model reflects its training data.

An AI model reflects its training data.

An AI model reflects its training data.

An AI model reflects its training data·

Figure 25: Data variants can be elusive

Anyone with a reasonable knowledge of the English language would conclude that all four sentences are the same. It may not occur to them that they are different fonts and styles unless they are specifically looking for such differences. For an AI model that performs optical character recognition (OCR), these four may be quite different from one another. What appear as subtle differences in images of an alphabet can sometimes make big difference to a model's efficacy or utility. A model trained with data comprising just one of these fonts may not perform well when tested with another.

It is tough for us to recognize which data variants are significantly distinct from a learning perspective. They may not always be as easy to identify as these well-distinguished fonts. For many of the AI tasks we are not proficient in identifying ML related data variations. This is especially true for the new tasks that we want the machines to learn about. For the well-known tasks that AI has already mastered, we know about most of the significant variations. However, such knowledge does not always help with new and niche tasks. Further, the significance of data variants also depends on the model's architecture and the learning algorithms used. This means that the task of identifying them would involve multiple people, like domain experts, architects and engineers. The entire team has to work together to achieve good data coverage for the AI models.

Data Variants and Model Development

The domain knowledge about data variants is important not just for data coverage but also for developing the AI models in the first place. Capturing intelligence with AI models is an unpredictable task, more so when we are trying something new without adequate references to follow. There are too many constituents in a learning exercise. We have to work iteratively on the model architecture, learning algorithm, training data, and the learning process itself. Often, this is a trial-and-error exercise, more so when we are using neural networks. When we embark on designing an AI model for a new task, it is good to start simple and add complexity progressively. This is where knowledge of data variants comes in handy. It helps in the progressive development and training of good models.

When we start with development of a new model, it is best to train it with just a few cohesive data variants, say just 4 out of the 12 available variants. This would reduce the training data needs, reduce demands on labeling, and keep the learning process relatively short. It would make it easier to fine-tune both the model architecture and the learning process. This is an iterative process and may require many iterations before we get good results. Once we are satisfied with the performance of the model, we can add the next level of complexity. This can be done by adding more variants to the training data. We can add variants v5 to v8 to the existing variants v1 to v4.

However, this is where the unpredictable nature of model development process manifests itself to the fore. The model’s performance may decline dramatically with introduction of additional data variants. What works for a few data variants may fall apart when more are introduced. The reason is that a model has a limited capacity to assimilate and encapsulate intelligence. This capacity is determined by the model architecture and the number of learnable parameters. When we increase the variants in data, we put additional demands on the model’s capacity. If we exceed this capacity, the training process will not yield desired results. The following diagram depicts this situation.

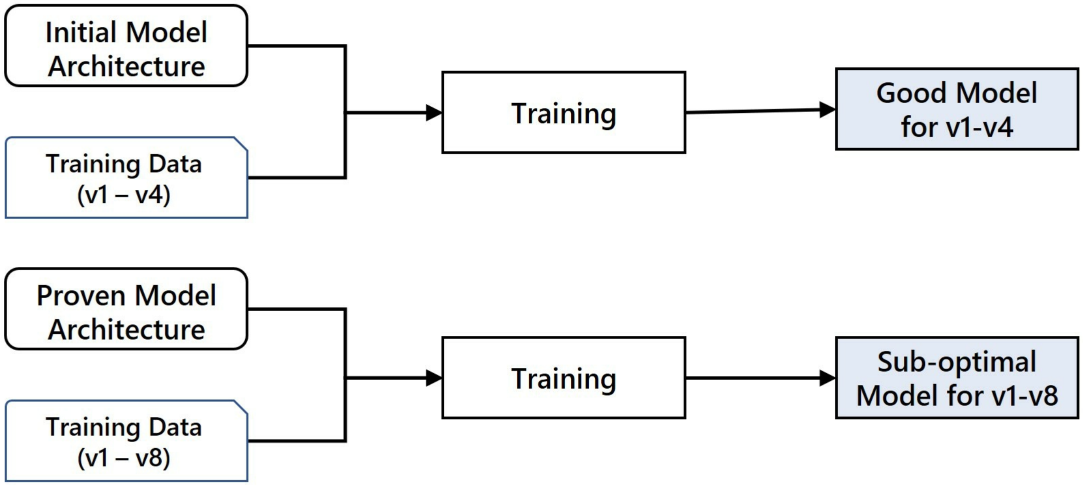


Figure 26: A proven architecture may fail with more data variants

With more data variants we often need a model with higher capacity. This is like going back to step 1, albeit with increased number of data variants. We would now require more training data and go through several iterations of the learning process. Yet it is not exactly back to step 1. We have one major advantage. We now start with an initial model architecture that worked well, at least for a few variants. Usually, we can enhance its capacity by increasing the number of learnable parameters, while retaining the architectural style. This would still make the architecture more complex. It would still require more training data and would be more complex to train. However, when it works, we would get a more capable model. This model would have wider applicability due to its better data coverage.

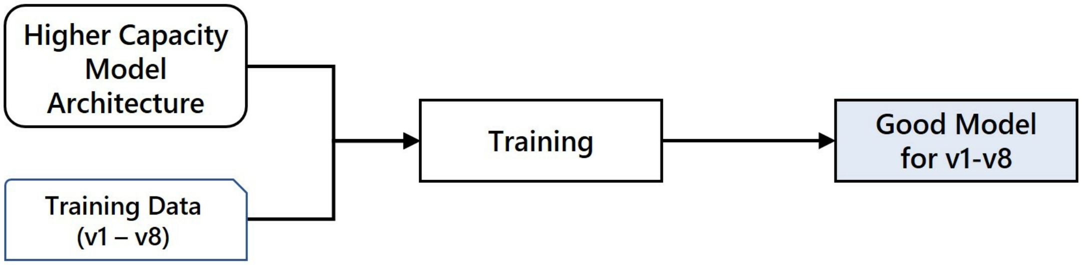


Figure 27: Larger number of variants may need a higher capacity model

However, things may not always work as described. In such a case, we would need to take a deeper look at the architecture and try the alternatives. We may have to revisit the kind of digitization we have used or the features we are using in the training data. Progress may require some innovative ways to mimic human thinking and utilize domain knowledge. Success may still elude us, and we may run out of time or budget to get the model right. Even in such situations we have one advantage. We have a model architecture that works for at least a few data variants, and we can use two approaches to leverage it.

We can reuse this working architecture only for the new set of variants. This would create multiple models within the domain, each catering to a different set of data variants. To accomplish this, we would require a good amount of domain knowledge. We have to choose the variants that are not just cohesive but also occur together in real life. For example, we can train an OCR model for just one or two fonts. However, if it has to be used on images that routinely contain several fonts, then the utility of the model will be quite low.

Sometimes, we can also leverage a trained neural network model, by retraining it with new data. We can take a model trained with variants v1 to v4, keep all the weights intact, and retrain it with data of variants v5 to v8. This way we not only reuse the architecture but also the learnable part of the model. This often works well when the underlying intelligence is of universal nature. It can speed up the training process and reduce the overall training data needs. The following diagram depicts the two approaches to leveraging a model architecture that has limited capacity.

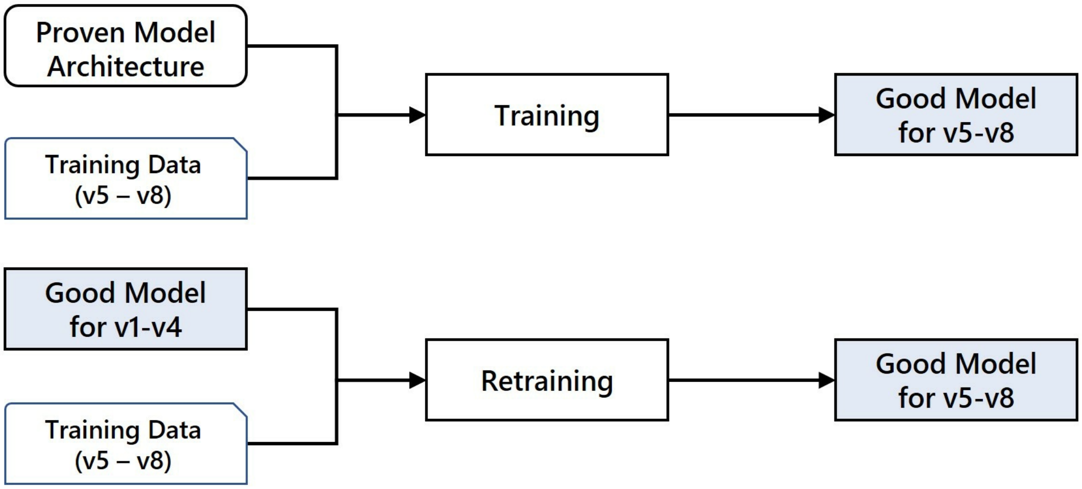
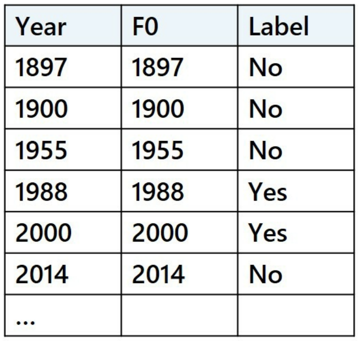


Figure 28: Two approaches to leverage model architectures

Feature Enrichment through Domain Insights

In the previous chapter, we used enriched features to train higher-level models. These enriched features were derived from pre-trained elementary AI models. This way, we used prior learnings for enriched features. There is another prolific source for enriched features. We can use our pre-existing knowledge to create enriched features. It can be as beneficial for model training as the use of prior learnings.

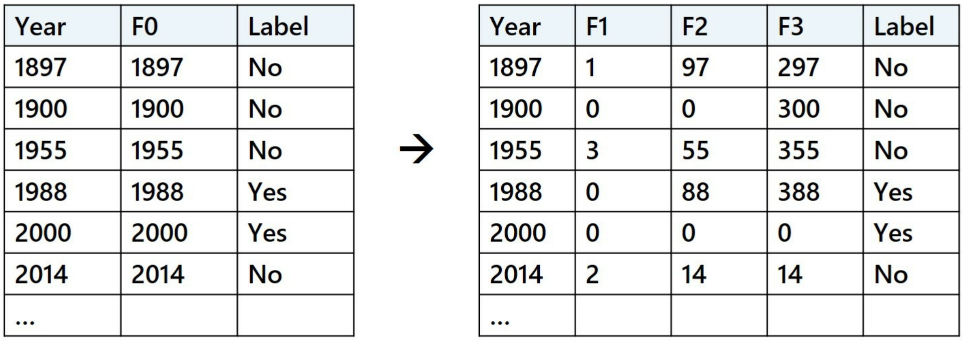
Let’s refer back to the leap year prediction task discussed in the previous section. Even though this task can easily be performed by a conventional program, let’s learn an AI model for it. What data do we use for training? The easiest approach is to provide the year number as the only feature with corresponding labels. This is what we did in an earlier section on ‘data coverage’, when we had the advantage of making assumptions about the efficacy of the learning algorithm. This simple training data is illustrated in the table below:



Year	F0	Label
1897	1897	No
1900	1900	No
1955	1955	No
1988	1988	Yes
2000	2000	Yes
2014	2014	No
…		

Figure 29: Basic training data for the leap year model

Given this single feature numeric data, what is the chance that a learning algorithm would figure out that division by 4 is the most critical rule for leap year predictions? The probability is not very high, more so when we provide limited amount of training data. The chances of learning the tougher 'division by 400' rule are close to zero. Can we help the machines learn better and faster? Can we tell them something that we already know? Like in this case, we know that the leap year prediction is about division by 4 or 400. How do we feed this to a reasonably standardized algorithm? How do we tell this to an algorithm that learns a decision tree or a neural network? One very effective way of doing so is to change the representation of the context. We can provide the pre-existing knowledge, through special features in the training data. For leap year prediction, this can be done by providing three special features about a year, as depicted below:



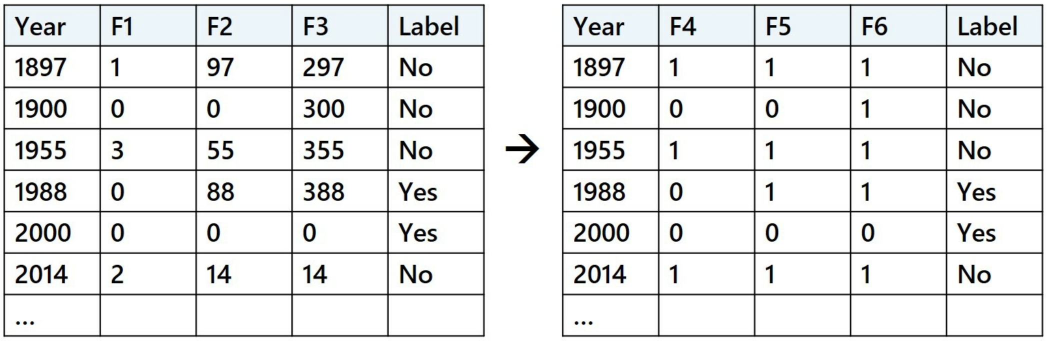
Year	F0	Label
1897	1897	No
1900	1900	No
1955	1955	No
1988	1988	Yes
2000	2000	Yes
2014	2014	No
…		

→

Year	F1	F2	F3	Label
1897	1	97	297	No
1900	0	0	300	No
1955	3	55	355	No
1988	0	88	388	Yes
2000	0	0	0	Yes
2014	2	14	14	No
…				

Figure 30: Improved training data with special features

The first feature ‘F1’ is a remainder after division by 4, and the other two are remainders after division by 100 and 400. With these new features we can do away with original feature ‘F0’. Learning about leap years now becomes easier. The feature ‘F1’ emerges as the most critical one for predictions. There are just four distinct values that it can take. It is easy to figure out the value that indicates a leap year. With such a useful feature in the data, many algorithms can create good models. However, we would still need a fair amount of data for century years, and yet the results may not be good. Can there be a better abstraction of training data, which would help the algorithms even more? Again, we can utilize our domain knowledge. The idea of a leap year relates to the existence of a remainder rather than its value. We can utilize this fact and rework the training data through some more preprocessing.



Year	F1	F2	F3	Label
1897	1	97	297	No
1900	0	0	300	No
1955	3	55	355	No
1988	0	88	388	Yes
2000	0	0	0	Yes
2014	2	14	14	No
...				

→

Year	F4	F5	F6	Label
1897	1	1	1	No
1900	0	0	1	No
1955	1	1	1	No
1988	0	1	1	Yes
2000	0	0	0	Yes
2014	1	1	1	No
...				

Figure 31: Feature enrichment by leveraging pre-existing knowledge

This new representation of a ‘year’ is even better for predicting a leap year. It passes on greater insights to a learning algorithm. It makes the task of learning simpler and more efficient too. Compared to feature ‘F1’, the feature ‘F4’ is an enriched feature. There are fewer distinct values that ‘F4’ can take when compared to ‘F1’. The enrichment in case of ‘F6’ over ‘F3’ is even better. The distinct values are reduced from 400 to just 2.

“When the features take fewer distinct values, the learning tends to become easier and requires less training data.”

In this leap year example, we have used mathematical pre-processing of data to provide better domain knowledge to learning algorithms. In case of Sports Eye’s live coverage system, we used elementary pre-trained models. Irrespective of the method used, we get better results through enriched features.

Embedding Domain Knowledge into the Model’s Architecture

Neural networks are a special class of AI models. They support numerous architectural styles and templates. With neural networks we can embed domain knowledge directly into the model’s architecture. The network can be designed to reflect the knowledge of our thinking process. This is a critical reason for neural networks to emerge as the model of choice for a very wide array of intelligence tasks.

When we use a neural network to recognize objects in an image, we usually provide pixel level data as input. For a monochrome image of just 40x40 pixels, we would provide 1600 features as input, each of which can take any of the 256 values. In a simple neural network architecture, the spatial information available in an image would get ignored. Even this can be quite successful at recognizing objects in images, but training such a model requires a lot of data. We can definitely do better. *We can utilize the fact that we see parts of an image with reference to its adjacent parts and not fully independent of them.* The spatial arrangements of pixels matter a lot for object recognition. To make learning more effective, the pixels must be seen in the context of adjacent pixels.

A Convolutional Neural Network (CNN) is an architectural style that takes cognizance of this spatial information. For example, a 3x3 convolution matrix tries to find the gist of 9 adjacent pixels at a time. A convolutional neural network does this for the entire image and over several layers. Without going into the mathematical details, we can conclude that a CNN lets us leverage the spatial significance of adjacent pixels for object recognition. Through convolution matrices, it extracts important characteristics and constituents of an image that contribute to object recognition. In a way, it reflects the human thinking process that leads us to object recognition. It embeds the role of spatial information for object recognition directly into the model architecture.

“Neural networks can be designed to imitate our thinking process, that too in an obscure digitized environment.”

CNNs are just one example of applying domain knowledge through model architecture. There are several neural network architectural styles that let us do the same in different domains. Recurrent neural networks (RNNs), long short-term memory (LSTMs) networks, attention-based transformers, etc., all imitate the human thinking process in one way or other. These neural network architectures help represent our mental patterns in digital space. As mentioned in Chapter 3, it is the beauty of the backpropagation algorithm that it works well with such varied architectural styles. This algorithm is indeed instrumental in leveraging our domain insights.

Summary

Knowledge of data variants is a critical piece of domain expertise that an AI team must have. It is a must to ensure data coverage, and it also guides the model development process.

Domain knowledge can be used to transform raw features into enriched ones. This would leverage pre-existing knowledge for easier training of AI models.

Neural networks provide the best support for embedding domain knowledge directly into the model architecture.

7 INSPIRING TRUST IN AI

Trust is a major area of concern with AI. As AI is being used for more and more tasks, this concern is also growing. There are many factors that contribute to the trust issues in AI. They range from obscurity of model architectures to the composition of training data. Also, there are several limitations in capturing intelligence through AI models. A trained model does not necessarily say ‘I don’t know’ when it encounters an unknown situation. It would still make a prediction or generate an output. This is an inherent property of most AI model, and it affects their usability and trustworthiness.

The trust issues play out differently for the creators and the end users of AI systems. The end users are at a significant disadvantage because they have limited access to information. They have limited or zero access to a model’s architecture, its training data and the learning process. Creators have more information and more tools at their disposal that help increase trust levels. However, trust issues remain, even for the creators of the system.

In this chapter, we explore multiple trust issues and highlight the caution areas. We also look at a few methods that would help increase trust in AI models.

Bring Along the Data

”In God we trust; all others must bring data.” This is a timeless quote from W. Edwards Deming, an authority in total quality management. The quote highlights the importance of data in the decision-making process. It advocates that decisions should be based on facts that are derived from real data. Above all, it reflects on the relationship between data and trust. Our trust in decisions increases when we understand the underlying data better. It increases when we know that the decision is based on real data. When we talk about trust in AI, the role of data is even more important. The quote by

Professor Deming can be rephrased for AI as follows:

“In God we trust, and AI models must disclose their training data.”

Most AI models are a product of training data. The intelligence that an AI model can encapsulate is defined by its architecture. However, the architecture on its own cannot capture the knowledge that is not available within the training data. *Just as people are known by the company they keep; the AI models are characterized by the data they learn from.* As we discussed in Chapter 6, the intelligence that an AI model actually captures depends on the data coverage by training data. It is determined by the data variants included in the training data. We can trust a model better when its training data is known and amenable to scrutiny.

For creators, these trust issues are largely about the undiscovered mismatch between the training data and the data on which the deployed model would be used. The data variants included in the two shall be similar for good results. For a model that has been pre-trained by others, we must have good information about its training data. Only then we can ascertain the possibility of this mismatch. For internally developed models the trust can increase through careful reviews of the data in the domain. Finally, creators do have the option of using the feedback from field data to improve the models. This helps in improving the model despite initial mismatch between training and field data. Thus, for creators, the data related issues are limited and fairly within their control.

For end users, the issues are more serious. The biggest threat is that of manipulation by model creators. When certain data variants are deliberately left out or overemphasized, the model is likely to perform for vested interests and not for the stated objectives. However, end-users do not have much information to assess this. The trust issues are therefore real for them, with limited options to improve the situation.

The impact of data issues differs as per the statistical sensitivity of AI models. When we learn a model to differentiate between a tiger and lion, the statistical sensitivity is low. If we require 100 images of each one, then adding another 500 just for lions is unlikely to make the model biased. However, this is not the case when we recommend a product based on the purchases of others. Here, the exclusion or overemphasis of certain kind of data would change the model’s output.

These issues are not peculiar to AI. They can be found in all decisions and predictions that are based on statistical data. The issues with data suppression or overemphasis are well known. *Mark Twain popularized the quote “Lies, damned lies, and statistics”, over a century ago.* With AI, the usage of statistically derived knowledge has increased, and its generation has been automated. As end users, we may have to find alternative means to figure out whether to trust a model or not. These may include the reputation of its creator, audit reports, and experiences of other users.

A Maze of Parameters

When a lending company or a bank receives a housing loan application, it faces a daunting task in approving it. Processing a loan application is a complex exercise for us humans. We cannot really take consistent decisions when we have to process a large number of parameters. As discussed in Chapter 4, the threshold for our working memory is quite low, just about 7 objects at a time. A loan application has hundreds of parameters. Yes, there are that many, though many of them are invisible to the applicant. An applicant’s credit history comprises hundreds of data points. The assessment of uncommitted monthly income requires processing of scores of bank transactions and other documents. The valuation of property is done on a large number of parameters. The application form itself has numerous fields.

A loan approval or denial easily refers to several hundred parameters. This raises two questions. First, how do lenders handle such a large number of parameters? Second, how do they ensure a reasonable degree of consistency in loan processing? The lenders use two solutions: enriched features and business rules.

When faced with a large number of decision parameters, we use tools, usually mathematical tools to reduce their numbers. Math is used to extract useful information from a number of parameters and put it into one or two of them. A credit score is a perfect example of this exercise. It takes a large amount of credit behavior data, and distills it into a credit score. To a loan officer this credit score reflects the applicant’s propensity to pay, and they can mostly avoid referring to the underlying data. The same is the case with computation of uncommitted monthly income. It summarizes information from several documents and months of transactions into a very useful number. This kind of information summarization is done for many parameters, and their numbers are reduced to more manageable levels.

However, despite all these efforts to reduce the number of parameters, they still remain well above the threshold of a person's processing capability. We need more to achieve consistency. This is where business rules come in handy. We have eliminator rules like - do not give a loan beyond 85% of the property value. Or, the monthly payment installments must not exceed x% of uncommitted monthly income. These rules do not reduce the number of parameters to be considered, but they tend to eliminate the cases that can cause business losses. They also ensure that the selected loan applications fall within some well-defined range for critical parameters. This reduces the variety that a loan officer has to handle. Here it must be noted, that the decision complexity is not just due to number of parameters but also due to the range of values a parameter can take. It is the latter that multiplies the number of valid combinations. A large number of valid and varied combinations put pressure on an officer's capability to take consistent decisions. The business rules reduce this pressure and make an officer's job more manageable.

Do we achieve consistency in loan processing through these two measures? Do we eliminate bias and errors? We succeed to a large extent. We do not achieve 100% consistency, but we do eliminate glaring errors. Bias too is reduced, especially for the cases that should not go through. The reason is that most of the rules are about elimination, like "do not give loan in x, y or z conditions." They rarely are about approving loans when x, y or z conditions are satisfied. Thus, the biases and errors of other kinds where a deserving loan is denied are tougher to address.

Can we trust an AI model that is learned from loan applications and the associated decisions? The question is more general though. Can we trust the models based on our labeling, where we ourselves are guided by business rules? Can we trust the consistency of our decisions when the number of underlying parameters is much higher than our processing capacity? The answer is – we cannot. We cannot be consistent with the decisions when it involves a large number of parameters. The decision making is guided by rules, and thereafter we invariably add our own biases. Learning from such data is likely to be suboptimal and would also carry the biases into the model. These limitations would be tough to detect, and even tougher to prove. Yet they exist. So as a thumb rule, we must be wary of AI models that relied on such labeling circumstances.

However, this does not mean that we should not use them. In fact, the models learned from such data are perhaps best way to upscale the expertise available with a few people. Taking decisions with large number of parameters requires experience and expertise. *There are so many dynamic objectives and inputs that everything cannot be made rule-based.* If it could be, we would not need to learn from data. We could just program these rules. The people taking these decisions are highly skilled in their work, and it is worth learning from their decisions. While we humans have our biases, we also have a special quality. Once there is awareness, we can consciously work to reduce and eliminate these biases. As a result, the decision-making keeps improving with time. All that is required is to remain cautious about potential biases and spread awareness within the team.

To make the best use of the intelligence and skills of the experts, we should treat their intelligence as dynamic intelligence. Accordingly, the AI models should be updated regularly to keep them in sync with the latest decision-making trends and imperatives. The decision-making role of humans should not be done away with completely. *Rather, it should be institutionalized carefully to power the up-scaling of expertise through AI.* The decision-making objectives and context are dynamic, and we need the experts to adjust well to new demands.

“To inspire trust, the AI models that encapsulate dynamic intelligence, should have a carefully configured ‘best before’ date.”

Obscurity of Model Architecture

In Chapter 3, we discussed two major classes of AI models, math-inspired and brain-inspired. It is relatively easy to trust something that is based on mathematics. Centuries of mathematics usage have strengthened this trust. It is not without reason. It is easy to see that 2 plus 6 is 8. A formula to compute average does compute average, and we understand what it means. Even when we do not understand the math behind a model, trust can still be there. We believe that the relevant mathematical foundations have been validated by knowledgeable people. We believe that the concepts used for the model have been validated by the right people, and are theoretically sound.

Should trust in math translate into trust in AI models that are math-oriented? The answer is both yes and no. Yes, because math does contribute to the robustness of a model and also provides insights into its working. We get a good idea of what kind of intelligence is being extracted from the data, what is its scope and how it may be used. The answer is also a ‘No’, because there is more to AI model than math. As discussed earlier, an AI model is a reflection of its training data. So, the data coverage issues discussed earlier would still apply to math-inspired models. However, once the data coverage and disclosure issues are addressed, math-oriented models would inspire a high level of trust.

The issues are quite different with neural network models. They support nearly free-form architecture, which can scale to millions of learnable parameters. There is no definitive way to understand how intelligence is represented in these neurons, layers, weights, and special variables. There are no established tools to help visualize the encapsulated intelligence. Naturally, it is tough to trust a model that we vaguely understand.

Neural networks inhibit trust due to their inscrutable ways of encapsulating intelligence. In some cases, we do have an idea of how it imitates our thinking process, but that may not be sufficient for inspiring trust. Unlike conventional code, which can be reviewed, we do not have review mechanisms for neural networks. A conventional code can also be analyzed at run time, but even if that is somewhat possible for neural networks, it may not be of much use. After all, we have a limited understanding of the structure itself, especially from a trust perspective.

Despite these issues, neural networks are very effective in capturing the extracted intelligence. They cannot be wished away for want of trust. We need to find alternative means to inspire trust. Comparing the architecture with well-established ones is one way. The disclosure of architecture, its lineage, and its prior successes for similar contexts would help. Reuse of an established architecture with new training data is even better. The disclosures of training data and the learning heuristics used during training, would also go a long way in inspiring trust. Evaluating these architectures, algorithms and data require special skills. The organizations creating AI systems must invest in acquiring and developing these skills. This would help them gain the trust of their end users. It would also help them build higher confidence in their own models.

Whose Choice is it?

Choice intelligence simultaneously operates in the realms of conventional programming and machine learning. It therefore suffers from two sources of distrust. For the part that is learned from data, we have trust issues based on training data and model architecture. The programmable part has a potential policy-driven bias.

For creators, the policy as well as the training data are under their control. They do not face much trust issues for choice intelligence, unless they stem from model architecture itself. Trust issues with choice intelligence are mainly for the end users. *For the end users, it is tough to know when a model switches between their interests and some vested interests.* Disclosures of training data, model architecture and policy would help to some extent. However, in most cases, the end users have to rely on the reputation and perceived intent of the creators.

Explain-ability and Trust

When we can explain our decisions, it increases trust in them. This applies not just to others but to us as well. We feel more confident about our own decisions when we have logical and rational explanations for them.

A lot of our explanations rely on logic, rules, and precedence. *Many times, the explanations are about the justification of a decision and not much about its correctness.* We cannot equate such explain-ability with trust. When we use precedence, we can always find instances that are aligned with our decisions. In reality, this may the mean suppression or overemphasis of certain data. However, these manipulations would be hidden, and bias would get legitimized. The same goes for logic and rules. We can cite the rules that favor our decisions and ignore the ones that do not.

Explain-ability, though a reasonable tool, is unlikely to add to trust in many circumstances. The bias and distrust scenarios discussed in the sections above cannot always be addressed through explanations. At times, precedence and rule-based explanations may be used to legitimize biases and vested interests. Such explanations may erode trust rather than strengthen it.

Summary

Disclosure of a model's training data can go a long way in inspiring trust.

The expertise of taking decisions on the basis of large number of parameters should be treated as dynamic intelligence. These AI models should be updated regularly to remain in sync with experts.

Math-oriented models are relatively easier to trust, especially with adequate data disclosure.

Despite obscurity of architecture, neural networks are indispensable. We have to use multiple methods to increase trust in them.

The solutions based on choice intelligence are most susceptible to manipulations, and would always have trust issues.

Explainable is not same as trustworthy. At times, explain-ability may be a fig leaf that hides biases and vested interests.

8 A META-TOOL

Are we more intelligent than the people who lived on this planet 100, 200 or 500 years ago? Has our average intelligence level risen dramatically in the last few generations? Being alive today, it is tempting to say yes. When we review the complexity of some of our present-day activities, the answer becomes an emphatic YES.

Most competitive sports today require far more skills than they would have when the modern Olympics started in 1896. The techniques, coaching, equipment, diet controls, etc., are all more sophisticated. It requires knowledge of many more disciplines to build a house today than it did about two centuries ago. It is more complex to build an LED bulb than a basic oil lamp. Kite surfing is more complicated than flying kites. It requires more skills today to drive a car than 100 years ago, simply because the speed has increased and traffic rules are more elaborate. In the medical field, modern day surgeries accomplish a lot more than the surgeries of yore.

It looks like we definitely have more knowledge and skills. Everything around us is more sophisticated today. But, is it really the only trend? Are we handling more complexity everywhere and for all kinds of activities? The answer is a clear no. Thanks to technology, so many things are getting simpler. When we took photographs with manual film cameras, we had to understand light conditions and the composition of the frame. We had to adjust the shutter speed, aperture, and focus accordingly. Nowadays, all these are automated. Most photography requires much fewer skills today. An automatic car is much easier to drive than one with gears. Further, the technologies like ESP, ABS, EBD, and now ADAS make it easier to drive successfully in tough circumstances. We may not always realize it, but many jobs are easier today than a few decades ago. We do not require more knowledge or skills for them.

Most of our progress over the last few generations is not because our raw intelligence is increasing. Rather, it is a result of the continuous application of the same level of intelligence. Each time we apply our brains to the current state of the art, we take things to the next level. *Initially, things become more capable and complex. Thereafter, we find ways to simplify them, while even adding to their capabilities.* The cumulative effects of this recurring process are astonishing. We are making exponential progress with practically the same level of general intelligence. With AI, computers are acquiring more and more of our skills and knowledge. Through the scaled application of this intelligence, they are now actively contributing to our progress. This contribution would only increase with time.

> *"Our progress is a result of cumulative application of our intelligence. AI would accelerate this process."*

A Tool for the Brain

In the physical world, we are able to make better use of our abilities with the help of various kinds of tools. A sophisticated tool significantly improves our capabilities in the physical world. We can see farther and clearer with binoculars, speak louder with microphones and speakers, move faster with cycles, and so on. With a car, we can move at over 100 km/h, and with a bullet train, we can move at over 300 km/h. X-ray and MRI machines improve our ability to diagnose ailments. Phones and internet improve our communication capabilities. The tools have been helping us from time immemorial; they are an integral part of our lives.

Such tools are not limited to physical world. Our general intelligence also gets enhanced by intellectual or mental tools. These mental tools make it easier for us to apply our raw intelligence. They improve the efficacy of our raw intelligence. Math is one such array of tools. Arithmetic, trigonometry, and calculus are all tools that help in disciplines as wide as physics, commerce, economics, sports, and more. Various engineering and scientific disciplines themselves are like advanced tools. They provide us with the conceptual frameworks required for innovations, discoveries, and inventions. They also provide the theoretical foundations for many of our activities and tasks. Any improvement or addition to these tools is a mark of intellectual progress. It lets us make better use of our raw intelligence, which in turn contributes towards accelerating our general progress.

AI is a special kind of intellectual tool. It is a sort of meta-tool. It has the special ability to make many of our intellectual tools available to all. The expertise that comes from years of training and practice can be made available to general public through AI. AI can diagnose diseases, approve loans, and provide legal advice. It can correct a yoga pose, coach a badminton player to serve better, and improve driving skills of a race car driver. AI is making the intellectual tools available to more and more people. This has profound implications for almost all the disciplines we aspire to specialize in. The impact on the education and training sectors cannot be overstated. The effect on the economy and the quality of life would be even bigger. *The upscaled access to intellectual tools would alter the trajectory of economic growth.* It would also redefine the potential of personal growth.

“AI is set to universalize the access to intellectual tools that otherwise require years of study, training and practice.”

Design AI, the AI for design tasks, would be a truly advanced tool. Currently, this kind of AI is in nascent stages. It helps us create new things, propose new solutions, and solve problems in creative ways. Soon, creativity would take on new avatars in many domains. As it progresses, the Design AI would work with our basic intelligence in manners that are tough to envisage today. Design AI may soon emerge as an intellectual tool in the league of engineering disciplines. It could add a new dimension to our general intelligence, and make it far more effective.

“AI is indeed a great opportunity to add to our collection of mental tools and disciplines.”

ACKNOWLEDGEMENT

Many thanks to all my friends who helped me with their reviews and insights.

ABOUT THE AUTHOR

Mukesh Borar

Mukesh is a technologist with expertise in designing advanced software systems. He works in the areas of artificial intelligence and software architecture.

Mukesh writes about AI and the technology's effects on different aspects of life. He lives in Bangalore, India. He likes travelling and is passionate about hiking in the high Himalayas.