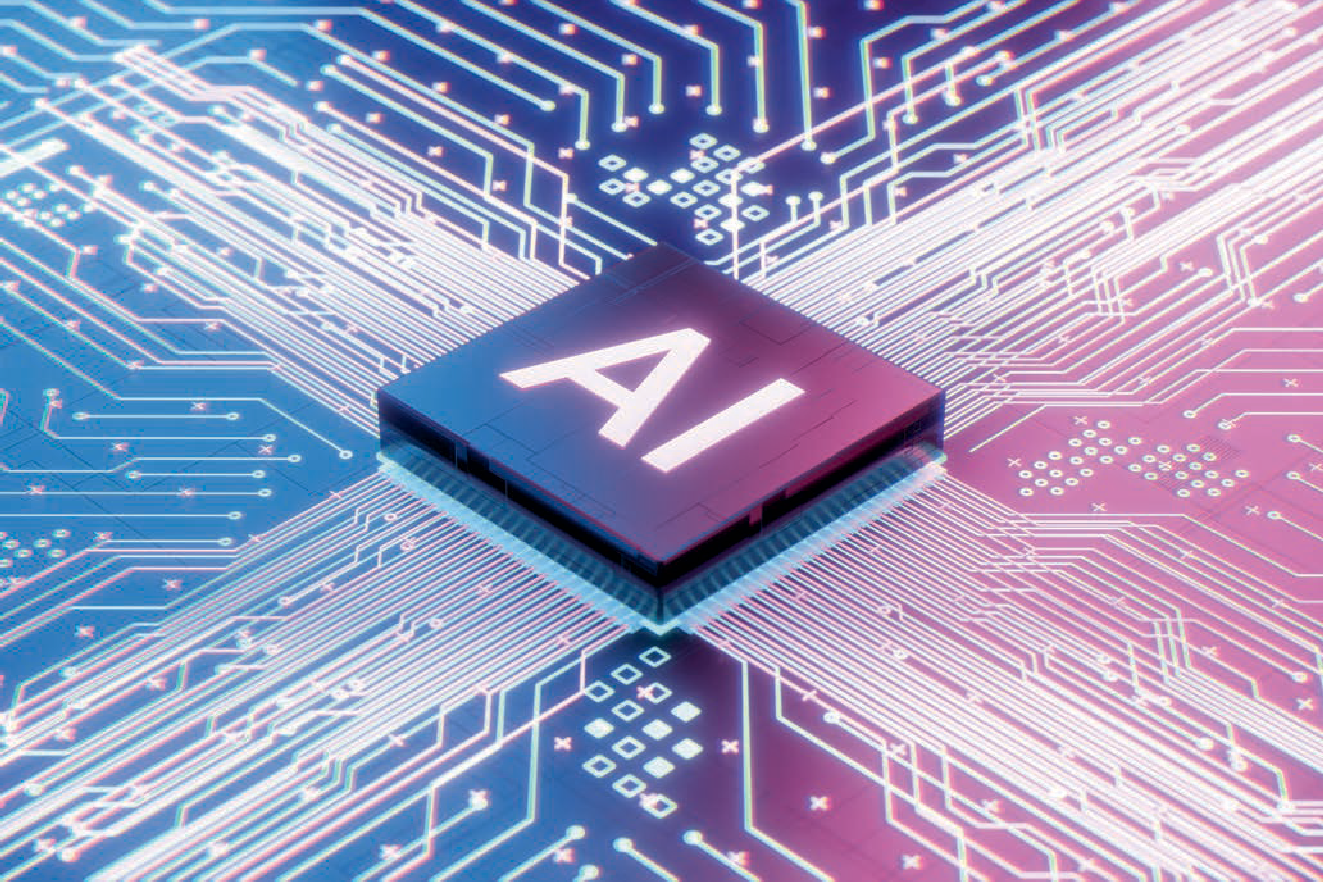
COURSE BOOK



## Artificial Intelligence

**DLBDSEAIS01**



Learning Objectives

##### Introduction **9**



In this course, you will get an introduction to the field of artificial intelligence.

The discipline of **Artificial Intelligence** originates from various fields of study such as cogni- tive science and neuroscience. The coursebook starts with an overview of important events and paradigms that have shaped the current understanding of artificial intelligence. In addi- tion, you will learn about the typical tasks and application areas of artificial intelligence.

On the completion of this coursebook, you will understand the concepts behind reinforce- ment learning, which are comparable to the human way of learning in the real world by exploration and exploitation.

Moreover, you will learn about the fundamentals of natural language processing and com- puter vision. Both are important for artificial agents to be able to interact with their environ- ment.



# Unit 1

## History of AI

#### STUDY GOALS

On completion of this unit, you will be able to …

… describe how artificial intelligence has developed as a scientific discipline.

… understand the different paradigms of artificial intelligence winter.

… explain the importance of expert systems and how they have contributed to artificial intelligence.

… talk about the advances of artificial intelligence.

DL-E-DLBDSEAIS01-U01

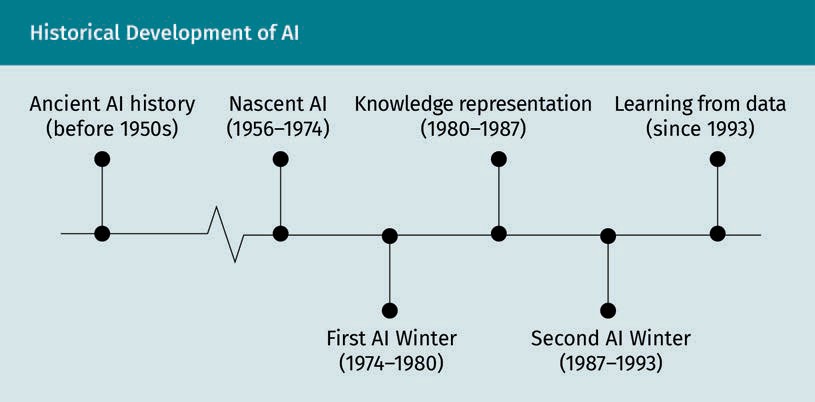
1. History of AI

### Introduction

This unit will discuss the history of artificial intelligence (AI). We will start with the his- torical developments of AI which date back to Ancient Greece. We will also discuss the recent history of AI.

In the next step, we will learn about the AI winters. From a historical perspective, there have been different hype cycles in the development of AI because not all requirements for a performant system could be met at that time.

We will also examine expert systems and their development. The last section closes with a discussion of the notable advances in artificial intelligence. This includes mod- ern concepts and its use cases.



The figure above illustrates the milestones in AI which will be discussed in the follow- ing sections.

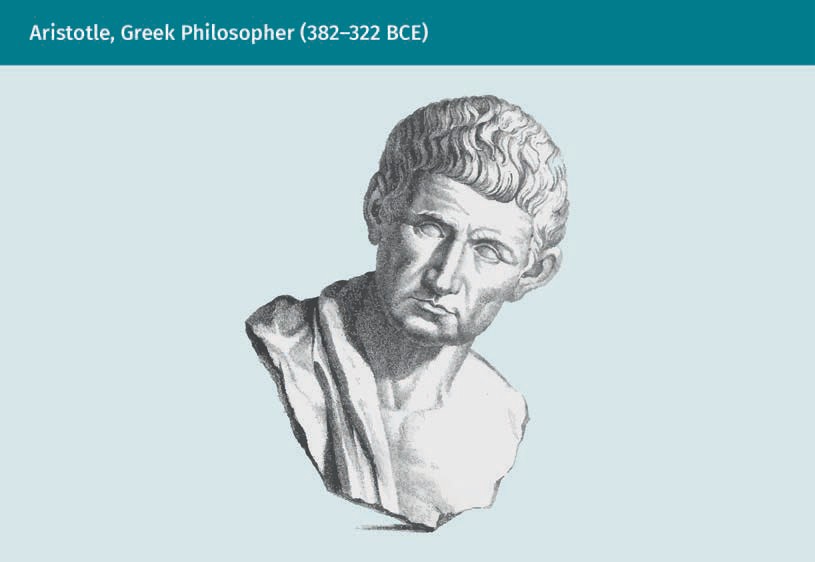
### Historical Developments

Even though historical views of artificial intelligence often start in the 1950s when it was first applied in computer science, the first considerations about AI range back to 350 BCE. Therefore, we will first start with a brief overview of ancient artificial intelli- gence history before we explore the more recent history.

History of AI

###### Aristotle, Greek Philosopher (384–322 BCE)

Aristotle was the first to formalize human thinking in a way to be able to imitate it. To formalize logical conclusions, he fully enumerated all possible categorical syllogisms (Giles, 2016).



Syllogisms (Greek: *syllogismós*, “conclusion”, “inference”) use deductive reasoning to derive workable conclusions from two or more given propositions. Logical programming languages as they are used today are based on a contemporary equivalent of Aristotle’s way to formalize thinking in the way logical derivations are used. Modern algorithms in AI can be programmed such that they derive valid logical conclusions based on a given set of previously defined rules.

###### Leonardo da Vinci, Italian Polymath (1452–1519)

Leonardo da Vinci designed a hypothetical computing machine on paper even though it was never put into practice. The machine had 13 registers, demonstrating that based on a stored program in memory or mechanics, a black box can accept inputs and pro- duce outputs.

These early considerations about computing machinery are very important because progress in computing is a necessary precondition for any sort of development in AI.

###### René Descartes, French Philosopher (1596–1650)

The French philosopher Descartes believed that rationality and reason can be defined using principles from mechanics and mathematics. The ability to formulate objectives using equations is an important foundation for AI, as its objectives are defined mathe- matically. According to Descartes, rationalism and materialism are two sides of the same coin (Bracken, 1984). This links to the methods used in AI where rational deci- sions are derived in a mathematical way.

###### Thomas Hobbes, British Philosopher (1588–1679)

Thomas Hobbes specified Descartes’ theories about rationality and reason. In his work, he identified similarities between human reasoning and computations of machines. Hobbes described that, in rational decision-making, humans employ operations similar to calculus, such that they can be formalized in a way that is analogous to mathematics (Flasiński, 2016).

###### David Hume, Scottish Philosopher (1711–1776)

Learning curve The learning curve is a graphical repre- sentation of the ratio between a learning outcome and the time required to solve a

new tasks.

Hume made fundamental contributions to questions of logical induction and the con- cept of causal reasoning (Wright, 2009). For example, he combined learning principles with repeated exposure, which has had – among others – a considerable influence on the **learning curve** (Russell & Norvig, 2022).

Nowadays, many machine learning algorithms are based on the principle of deriving patterns or relations in data through repeated exposure.

###### Recent History of Artificial Intelligence

The recent history of AI started around 1956 when the seminal Dartmouth conference took place. The term artificial intelligence was first coined at this conference and a defi- nition of the concept was proposed (Nilsson, 2009). In the following, we will discuss the the key personalities, organizations, and concepts in the development of AI.

Key personalities

The recent history of AI normally starts with the pioneering Dartmouth conference in 1956 where the term ‘artificial intelligence’ was first coined, and a definition of the term was suggested.

During the decade of AI's inception, important personalities contributed to the disci- pline.

History of AI

Alan Turing was an English computer scientist and mathematician who formalized and mechanized rational thought processes. In 1950 he conceptualized the well-known Turing Test. This test examines if an AI communicates with a human observer without the human observer being able to distinguish whether they are conversing with a machine or another human. If the human cannot identify an AI as such, it is considered a real AI (Turing, 1950).

The American scientist John McCarthy studied automata. It was he who first coined the term “artificial intelligence” during preparations for the Dartmouth conference (McCar- thy *et al*., 1955). In cooperation with the Massachusetts Institute of Technology (MIT) and International Business Machines (IBM), he established AI as an independent field of study. He was the inventor of the programming language Lisp in 1958 (McCarthy, 1960). For more than 30 years LISP was used in a variety of applications of AI, such as fraud detection and robotics. In the 1960s, he founded the Stanford Artificial Intelli- gence Laboratory which has had a significant influence on research on implementing human capabilities, like reasoning, listening, and seeing, in machines (Feigenbaum, 2012).

American researcher Marvin Minsky, a founder of the MIT Artificial Intelligence Labora- tory in 1959, was another important participant in the Dartmouth conference. Minsky combined insights from AI and cognitive science (Horgan, 1993).

With a background in linguistics and philosophy, Noam Chomsky is another scientist who contributed to the development of AI. His works about formal language theory and the development of the Chomsky hierarchy still play an important role in areas such as natural language processing (NLP). Besides that, he is well known for his critical views on topics such as social media.

Key institutions

The most influential institutions involved in the development of AI are Dartmouth Col- lege and MIT. Since the Dartmouth conference, there have been several important con- ferences at Dartmouth College discussing the latest developments in AI. Many of the early influential AI researchers have taught at MIT, making it a key institution for AI research. But also companies such as IBM and Intel, and government research institu- tions, such as the Defense Advanced Research Projects (DARPA), have contributed much to AI by funding research on the subject (Crevier, 1993).

Key disciplines contributing to the development of AI

Many research areas have been contributing to the development of artificial intelli- gence. The most important areas are decision theory, game theory, neuroscience, and natural language processing:

* In decision theory mathematical probability and economic utility are combined. This provides the formal criteria for decision-making in AI regarding economic benefit and dealing with uncertainty.
* Game theory is an important foundation for rational agents to learn strategies to solve games. It is based on the research of the American–Hungarian computer sci- entist John von Neuman (1903–1957), and the American–German mathematician and game theorist Oskar Morgenstern (1902–1977); (Leonard, 2010).
* The insights from neuroscience about how the brain works are increasingly used in artificial intelligence models, especially as the importance of artificial neural net- works (ANN) is increasing. Nowadays, there are many models in AI trying to emulate the way the brain stores information and solves problems.
* Natural language processing (NLP) combines linguistics and computer science. The goal of NLP is to process not only written language (text) but also spoken language (speech).

High-level programming languages are important to program AI. They are closer to human language than low-level programming languages such as machine code or assembly language and allow programmers to work independently from the hardware’s instruction sets. Some of the languages that have been developed specifically for AI are Lisp, Prolog, and Python:

* Lisp has been developed by John McCarthy and is one of the oldest programming languages. The name comes from ‘list processing’ as Lisp is able to process charac- ter strings in a unique way (McCarthy, 1960). Even though it dates back to the 1960s it has not only been used for early AI programming but is still relevant today.
* Another early AI programming language is Prolog which was specially designed to prove theorems and solve logical formulas.
* Nowadays, the general-purpose high-level programming language Python is the most important programming language. As Python is open source, there exist exten- sive libraries which help programmers to create applications in a very efficient way.

There are three important factors that have contributed to the recent progress in artifi- cial intelligence:

* Increasing availability of massive amounts of data, which are required to develop and train AI algorithms.
* Large improvements in data processing capacity of computers.
* New insights from mathematics, cognitive science, philosophy, and machine learn- ing.

These factors support the development of approaches that were previously impossible, be it because of a lack of processing capability or a lack of training data.

History of AI

### AI Winter

The term “AI winter” first appeared in the 1980s. It was coined by AI researchers to describe periods when interest, research activities, and funding of AI projects signifi- cantly decreased (Crevier, 1993). The term might sound a bit dramatic. However, it reflects the culture of AI, which is known for its excitement and exuberance.

Historically, the term has its origin in the expression “nuclear winter”, which is an after- effect of a hypothetical nuclear world war. It describes the state where the atmosphere is overcome by ashes and the sunshine cannot reach the Earth’s atmosphere, meaning that temperatures would drop excessively and nothing would be able to grow. There- fore, transferring this term to AI, it marks periods where interest and funding of AI tech- nologies were significantly reduced, causing a reduction in research activities. Down- turns like this are usually based on exaggerated expectations towards the capabilities of new technologies that cannot be realistically met.

There have been two AI winters. The first lasted approximately from 1974 to 1980 and the second from 1987 to 1993 (Crevier, 1993).

###### The First AI Winter (1974–1980)

During the cold war between the former Soviet Union and the United States (US), auto- matic language translation was one of the major drivers to fund AI research activities (Hutchins, 1997). As there were not enough translators to meet the demand, expecta- tions were high to automate this task. However, the promised outcomes in machine translation could not be met. Early attempts to automatically translate language failed spectacularly. One of the big challenges at that time was handling word ambiguities. For instance, the English sentence “out of sight, out of mind” was translated into Rus- sian as the equivalent of “invisible idiot” (Hutchins, 1995).

When the Automatic Language Processing Advisory Committee evaluated the results of the research that had been generously funded by the US, they concluded that machine translations are not as accurate, nor faster nor cheaper than employing humans (Auto- matic Language Proesccing Advisory Committee, 1966). Additionally, perceptrons – which were at that time a popular model of neural-inspired AI – had severe shortcom- ings as even simple logical functions, such as exclusive or (XOR), could not be repre- sented in those early systems.

###### The Second AI Winter (1987–1993)

The second AI winter started around 1987 when the AI community became more pessi- mistic about developments. One major reason for this was the collapse of the **Lisp machine** business which led to the perception that the industry might end (Newquist, 1994). Moreover, it turned out that it was not possible to develop early successful exam- ples of expert systems beyond a certain point. Those expert systems had been the

Lisp machine A Lisp machine is a type of computer that supports the Lisp language.

main driver of the returned interest in AI systems after the first AI winter. The reason for the limitations was that the growth of fact databases was no longer manageable, and results were unreliable towards unknown inputs i.e., inputs on which the machines had not been trained.

However, there are also arguments that there are no such thing as AI winters, and that they are myths spread by a few prominent researchers and organizations who had lost money (Kurzweil, 2014). While the interest in Lisp machines and expert systems decreased, AI was still deeply embedded in many other types of processing operations such as credit card transactions.

###### Causes of the AI Winters

There are several conditions that can cause AI winters. The three most important requirements for the success of artificial intelligence are

* algorithms and experience with them,
* computing capacity, and
* the availability of data.

The past AI winters occurred because not all requirements were met.

During the first AI winter, there were already powerful algorithms. However, for success- ful results, it is necessary to process a huge amount of data. This requires a lot of memory capacity as well as high processing speed. At the time, there were not enough data available to properly train those algorithms. Therefore, the expectations of inter- ested parties and investors could not be met. As the funded research was unable to produce the promised results, the funding was stopped.

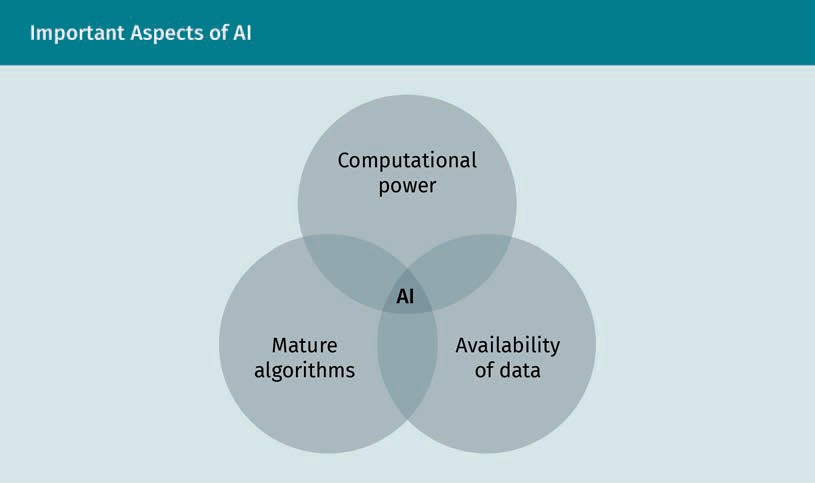
Until the 1980s the computing capacity had increased enough to train the available algorithms on small data sets. However, as approaches from machine learning and deep learning became integral parts of AI in the late 1980s, there was a greater need for large data sets to train AI systems, which became an issue. The lack of labeled train- ing data – even though computing capacity would have been available – created the perception that several of the AI projects had failed.

As the AI winters show, it is impossible to make progress towards developing algo- rithms for AI unless there is enough computing capacity (i.e., data storage and process- ing speed) and training data.

###### The Next AI Winter

Nowadays, all three aspects mentioned above are fully met. There is enough computa- tional power to train the available algorithms on a large number of existing data sets. The figure below summarizes the preconditions for AI to be successful.

History of AI



However, the question of whether there might be another AI winter in the future can hardly be answered. If a hyped concept gets a lot of funding but does not perform, it might be defunded which could cause another AI winter. Nevertheless, nowadays AI technologies are embedded in many other fields of research. If low-performing projects are defunded, there is always room for new developments. Therefore, everybody is free to decide whether AI winters are simply a myth or if the concept really matters.

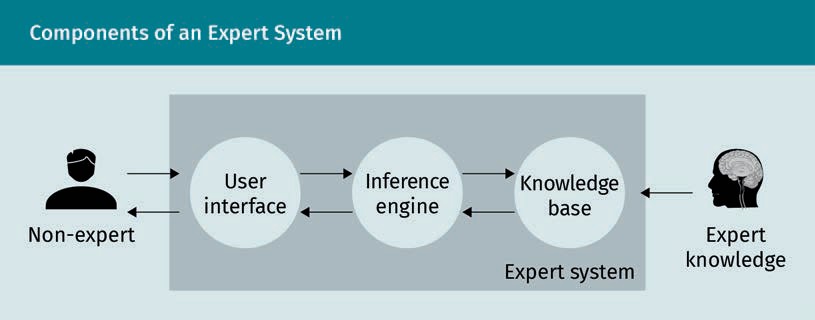
### Expert Systems

One of the key concepts when looking at the history of artificial intelligence are expert systems. Expert systems belong to the group of knowledge-based systems. As the name suggests, the goal of expert systems is to emulate the decision and solution-finding process using the domain-specific knowledge of an expert. The word “expert” refers to a human with specialized experience and knowledge in a given field, such as medicine or mechanics. Since problems in any given domain may be similar to each other, but never quite alike, solving problems in that domain cannot be accomplished by memori- zation alone. Rather, problem-solving is supplemented by a method that involves matching or applying experiential knowledge to new problems and application scenar- ios.

###### Components of an Expert System

Expert systems are designed to help a non-expert user make decisions based on the knowledge of an expert.

The figure below illustrates the typical components of an expert system:



Expert systems are composed of a body of formalized expert knowledge from a specific application domain, which is stored in the knowledge base. The inference engine uses the knowledge base to draw conclusions from the rules and facts in the knowledge. It implements rules of logical reasoning to derive new facts, rules, and conclusions not explicitly contained in the given corpus of the knowledge base. A user interface enables the non-expert user to interact with the expert system to solve a given problem from the application domain.

###### Types of Expert Systems

With respect to the representation of knowledge, three approaches to expert systems can be distinguished:

* Case-based systems store examples of concrete problems together with a successful solution. When presented with a novel, previously unseen case, the system tries to retrieve a solution to a similar case and apply this solution to the case at hand. The key challenge is defining a suitable similarity measure to compare problem settings.
* Rule-based systems represent the knowledge base in the form of facts and if-A- then-B-type rules that describe relations between facts.
* If the problem class to be solved can be categorized as a decision problem, the knowledge can be represented in a decision tree. The latter are typically generated by analyzing a set of examples.

###### Development of Expert Systems

Historically, expert systems are an outgrowth of earlier attempts at implementing a general problem solver. This approach is primarily associated with the researchers Her- bert A. Simon and Allen Newell, who, in the late 1950s, used a combination of insights from cognitive science and mathematical models of formal reasoning to build a system intended to solve arbitrary problems by successive reduction to simpler problems (Kuipers & Prasad, 2021). While this attempt was ultimately considered a failure when compared to its lofty goals, it has nevertheless proven highly influential in the develop- ment of cognitive science.

History of AI

One of the initial insights gained from the attempt at general problem solving was that the construction of a domain specific problem solver should—at least in principle—be easier to achieve. This led the way to think about systems that combined domain-spe- cific knowledge with domain-dependent apposite reasoning patterns. Edward Feigen- baum, who worked at Stanford University, the leading academic institution for the sub- ject at the time, defined the term expert system and built the first practical examples while leading the Heuristic Programming Project (Kuipers & Prasad, 2021).

The first notable application was Dendral, a system for identifying organic molecules. In the next step, expert systems were established to help with medical diagnoses of infec- tious diseases based on given data and rules (Woods, 1973). The expert system that evolved out of this was called MYCIN, which had a knowledge base of around 600 rules. However, it took until the 1980s for expert systems to reach the height of research interest, leading to the development of commercial applications.

The main achievement of expert systems was their role in pioneering the idea of a for- mal, yet accessible representation of knowledge. This representation was explicit in the sense that it was formulated as a set of facts and rules that were suitable for creation, inspection, and review by a domain expert. This approach thus clearly separates domain-specific business logic from the general logic needed to run the program – the latter encapsulated in the inference engine. In stark contrast, more conventional pro- gramming approaches implicitly represent both internal control and business logic in the form of a program code that is hard to read and understand by people who are not IT experts. At least in principle, the approach championed by expert systems enabled even non-programmers to develop, improve, and maintain a software solution. More- over, it introduced the idea of rapid prototyping since the fixed inference engine ena- bled the creation of programs for entirely different purposes simply by changing the set of underlying rules in the knowledge base.

However, a major downside of the classical expert system paradigm, which also finally led to a sharp decline in its popularity, was also related to the knowledge base. As expert systems were engineered for a growing number of applications, many interesting use cases required larger and larger knowledge bases to satisfactorily represent the domain in question. This insight proved problematic in two different aspects:

1. Firstly, the computational complexity of inference grows faster than it does linearly in the number of facts and rules. This means that for many practical problems the system’s answering times were prohibitively high.
2. Secondly, as a knowledge base grows, proving its consistency by ensuring that no constituent parts contradict each other, becomes exceedingly challenging.

Additionally, rule-based systems in general lack the ability to learn from experience. Existing rules cannot be modified by the expert system itself. Updates of the knowledge base can only be done by the expert.

### Notable Advances

After illustrating the downturns of AI winters, it is time to shift the focus to the prosper- ous times when artificial intelligence has made huge advances. After an overview of the research topics that have been in focus in the respective eras, we will examine the most important developments in adjacent fields of study and how they relate to the progress in artificial intelligence. Finally, we will examine the future prospects of AI.

###### Nascent Artificial Intelligence (1956–1974)

In the early years, AI research was dominated by the “symbolic” AI. In this approach, rules from formal logic are used to formalize thought processes as manipulation of symbolic representations of information. Accordingly, AI systems developed during this era deal with the implementation of logical calculus. In most cases, this is done by implementing a search strategy, where solutions are derived in a step-by-step proce- dure. The steps in this procedure are either inferred logically from a preceding step or systematically derived using backtracking of possible alternatives to avoid dead ends.

The early years were also the period where first attempts for natural language process- ing were developed. The first approaches for language processing were focused on highly limited environments and settings. Therefore, it was possible to achieve initial successes. The simplification of working environments – a “microworld” approach – also yielded good results in the fields of computer vision and robot control.

In parallel, the first theoretical models of neurons were developed. The research focus was on the interaction between those cells (i.e., computational units) to implement basic logical functions in networks.

###### Knowledge Representation (1980–1987)

The focus of the first wave of AI research was primarily on logical inference. In contrast, the main topics of the second wave were driven by the attempt to solve the problem of knowledge representation. The reason for this focus shift was caused by the insight that in day-to-day situations intelligent behavior is not only based on logical inference but much more on general knowledge about the way the world works. This knowledge- based way to view intelligence was the origin of early expert systems. The main charac- teristic of these technologies was that domain-relevant knowledge was systematically stored in databases. Using these databases, a set of methods was developed to access that knowledge in an efficient, effective way.

The emerging interest in AI after the first AI winter was also accompanied by an upturn in governmental funding at the beginning of the 1980s with projects such as the Alvey project in the UK and the Fifth Generation Computer project of the Japanese Govern- ment (Russell & Norvig, 2022).

History of AI

Additionally, in this period the early throwbacks of neurally-inspired AI approaches could be addressed by new network models and the use of backpropagation as a train- ing method in layered networks of computational units.

###### Learning from Data (Since 1993)

During the 1990s there were some major advances of AI in games when the first com- puter system “Deep Blue” was able to beat Garry Kasparov, the world champion in chess at that time.

Aside from this notable but narrow success, AI methods have become widely used in the development of real-world applications. Successful approaches in the subfields of AI have gradually found their way into everyday life – often without being explicitly labeled as AI. In addition, since the early 1990s, there has been a growing number of ideas from decision theory, mathematics, statistics, and operations research that those contributed significantly to AI becoming a rigorous and mature scientific discipline. Especially the paradigm of intelligent agents has become increasingly popular. In this context, the concept of intelligent agents from economic theory combines with the notions of objects and modularity of computer science and forms the idea of entities that can act intelligently. This perspective allows it to shift perspective from AI being an imitation of human intelligence to the study of intelligent agents and a broader study of intelligence in general.

The advances in AI since the 1990s have been supported by a significant increase in data storage and computational capacities. Along with this, during the rise of the inter- net, there has been an incomparable increase in variety, velocity, and volume of gener- ated data, which also supported the AI boom.

In 2012 the latest upturn in the interest of AI research started when deep learning was developed based on advances in connectionists machine learning models. The increase in data processing and information storage capabilities combined with larger data corpora brought theoretical advances in machine learning models into practice. With deep learning, new performance levels in many machine learning benchmark problems could be achieved. This led to a revival of interest in well-established learn- ing models, like reinforcement learning and created space for new ideas, like adversa- rial learning.

###### Adjacent Fields of Study

There are many fields of study that continuously contribute to AI research. The most influential fields will be described in the following.

Linguistics

Linguistics can be broadly described as the science of natural language. It deals with exploring the structural (grammatical) and phonetic properties of interpersonal com- munication. To understand language, it is necessary to understand the context and the

subject matter in which it is used. In his book *Syntactic Structures*, Noam Chomsky (1957) made an important contribution to linguistics and, therefore, to natural language processing. Since our thoughts are so closely linked to language as a form of represen- tation, one could take it a step further and link creativity and thought to linguistic AI. For example, how is it possible that a child says something it has never said before? In AI, we understand natural language as a medium of communication in a specific con- text. Therefore, language is much more than just a representation of words.

Cognition

In the context of AI, cognition refers to different capabilities such as perception and cognition, reasoning, intelligence, learning and understanding, and thinking and com- prehension. This is also reflected in the word “recognition”. A large part of our current understanding of cognition is a combination of psychology and computer science. In psychology, theories and hypotheses are formed from observations with humans and animals. In computer science, behavior is modeled based on what has been observed in psychology. When modeling the brain by a computer, we have the same principle of stimulus and response as in the human brain. When the computer receives a stimulus, an internal representation of that stimulus is made. The response to that stimulus can lead to the original model being modified. Once we have a well-working computer model for a specific situation, the next step will be to find out how decisions are made. As decisions based on AI are involved in more and more areas of our lives, it is impor- tant to have high transparency about the reasoning process to an external observer. Therefore, explainability (the ability to explain, how a decision has been made) is becoming increasingly important. However, approaches based on deep learning still lack explainability.

Games

When relating games to AI, this includes much more than gambling or computer games. Rather, games refer to learning, probability, and uncertainty. In the early twentieth cen- tury, game theory was established as a mathematical field of study by Oskar Morgen- stern and John von Neuman (Leonard, 2010). In game theory, a comprehensive taxon- omy of games was developed and, in connection with this, some gaming strategies that have been proven to be optimal strategies.

Another discipline related to game theory is decision theory. While game theory is more about how the moves of one player affect the options of another player, decision theory deals with usefulness and uncertainty, i.e., utility and probability. Both are not necessarily about winning but more about learning, experimenting with possible options, and finding out what works based on observations.

Games, like chess, checkers, and poker, are usually played for the challenge of winning or for entertainment. Nowadays, machines can play better than human players. Until 2016, people believed that the game of Go might be an unsolvable challenge for com- puters because of its combinatorial complexity. The objective of the game is to sur- round the most territory on a board with 19 horizontal and vertical lines. Even though the ruleset is quite simple, the complexity comes from the large size of the game board and the resulting number of possible moves. This complexity makes it impossible to apply methods that have been used for games like chess and checkers. However, in

History of AI

2015 DeepMind developed the system AlphaGo based on deep networks and reinforce- ment learning. This system was the first to be able to beat Lee Sedol, one of the world’s best Go players (Silver *et al*., 2016).

Not long after AlphaGo, DeepMind developed the system AlphaZero (Silver *et al*., 2018). In contrast to AlphaGo, which learned from Go knowledge from past records, AlphaZero only learns based on intensive self-play following the set of rules. This system turned out to be even stronger than AlphaGo. It is also remarkable that AlphaZero even found some effective and efficient strategies, which had, so far, been missed by Go experts.

The Internet of Things

It has only been a few years since the term “Internet of things” (IoT) first came up. IoT connects physical and virtual devices using technologies from information and com- munication technology. In our everyday lives, we are surrounded by a multitude of physical devices that are always connected, such as phones, smart home devices, cars, and wearables. The communication between those devices produces a huge amount of data which links IoT to AI. While IoT itself is only about connecting devices and collect- ing data, AI can help add intelligent behavior to the interaction between those machines.

Having intelligent devices integrated into our everyday lives not only create opportuni- ties but also many new challenges. For instance, data about medication based on phys- ical measurements of a wearable device could be used positively, to remind a person about medication intake, but also to decide about a possible increase in their health insurance rate. Therefore, topics like ethics of data use and privacy violations, become increasingly important facing the new fields of use of AI.

Quantum computing

Quantum computing is based on the physical theory of quantum mechanics. Quantum mechanics deal with the behavior of sub-atomic particles which follow different rules than described by theories from classical physics. For instance, in quantum mechanics, it is possible that an electron can be in two different states at the same time. Quantum mechanics assumes that physical systems can be characterized using a wave function describing the probabilities of the system being in a particular state. The goal is to exploit these quantum properties to build supercomputers where new algorithmic approaches can be implemented, allowing them to outperform classical machines (Giles, 2018). The kind of information processing from quantum computing is well suited for the probabilistic approach which is inherent in many AI technologies. There- fore, quantum computers offer the possibility of accelerating applications with AI and thus achieve a real advantage in processing speed. However, due to the early stage of development of these systems, using quantum computing has hardly been researched.

###### The Future of AI

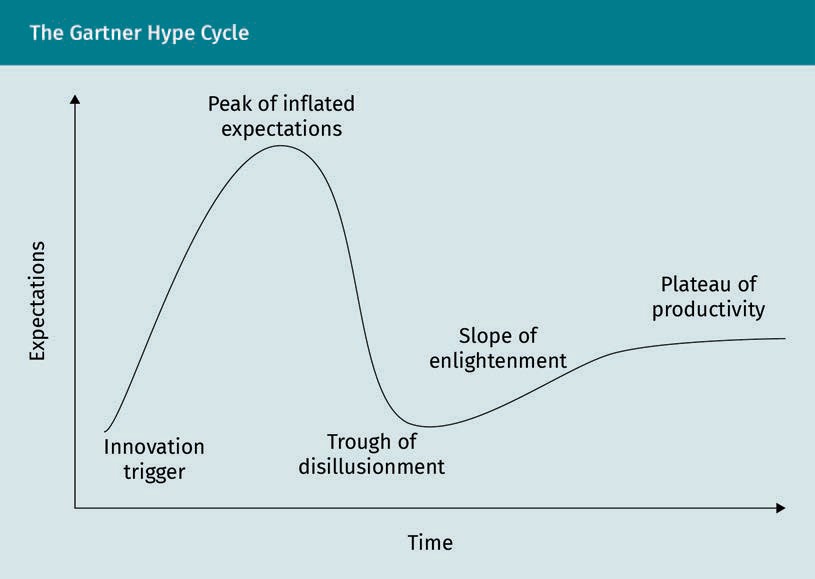
It is always highly speculative when trying to assess the impact of a research area or new technology on the future as the future prospects will always be biased by previous experiences. Therefore, we do not attempt to predict the long-term future of AI. Never- theless, we want to examine the directions of developments in AI and the supporting technologies.

The Gartner hype curve is frequently used to evaluate the potential of new technolo- gies (Gartner, 2021). The hype curve is presented in a diagram where the y-axis repre- sents the expectations towards a new technology and time is plotted on the x-axis.

The time axis is characterized by five phases:

1. In the discovery phase a technological trigger or breakthrough generates significant interest and triggers the innovation.
2. The peak phase of exaggerated expectations is usually accompanied by much enthusiasm. Even though there may be successful applications most of them strug- gle with early problems.
3. The period of disillusionment shows that not all expectations can be met.
4. In the period of enlightenment, the value of innovation is recognized. There is an understanding of the practical understanding and advantages, but also of the limi- tations of the new technology.
5. In the last period, a plateau of productivity is reached, and the new technology becomes the norm. The final level of this plateau depends on whether the technol- ogy is adopted in a niche or a mass market.

History of AI



The hype cycle has some similarities with the inverted U-shape of a normal distribu- tion except that the right end of the curve leads into an increasing slope that eventu- ally flattens out.

In 2021, the hype cycle for artificial intelligence showed the following trends (Gartner, 2021):

* In the innovation trigger phase, subjects like composite AI (a combination of differ- ent approaches from AI) and general AI (the ability of a machine to perform human- like intellectual tasks) appear. Moreover, topics like Human-Centered AI and Respon- sible AI show that human integration is becoming increasingly important for the future of AI.
* Deep neural networks, which have been the driver for new levels of performance in many machine learning applications over the past decades, are still at the peak phase of inflated expectations or hype. Moreover, topics like knowledge graphs and smart robots appear in that phase.
* In the disillusionment phase, we find topics like autonomous vehicles, which have experienced defunding as the high expectations in this area could not be met.

So far, none of the topics of AI have yet reached the plateau of productivity. This reflects the general acceptance of this area and the productive use of the related tech- nologies.

**Summary**

Research about artificial intelligence has been of interest for a long time. The first theoretical thoughts about artificial intelligence date back to Greek philosophers like Aristotle. Those early considerations were continued by philosophers like Hobbes and Descartes. Since the 1950s, it has also become an important compo- nent of computer science and made important contributions in areas such as knowledge representation in expert systems, machine learning, and modeling neu- ral networks.

In the past decades, there have been several ups and downs in AI research. They were caused by a cycle between innovations accompanied by high expectations and disappointment when those expectations could not be met, often because of technical limitations.

Over time, AI has been shaped by different paradigms from multiple disciplines. The most popular paradigm nowadays is deep learning. New fields of applications like IoT or quantum computing offer a vast amount of opportunities of how AI can be used. However, it remains to see how intelligent behavior will be implemented in machines in the future.



# Unit 2

## Modern AI Systems

#### STUDY GOALS

On completion of this unit, you will be able to…

… explain the difference between narrow and general artificial intelligence systems.

… name the most important application areas for artificial intelligence.

… understand the importance of artificial intelligence for corporate activities.

DL-E-DLBDSEAIS01-U02

1. Modern AI Systems

### Introduction

Artificial intelligence has become an integral part of our everyday life. There are several examples where we do not even notice the presence of AI, be it in Google maps or smart replies in Gmail.

There are two categories of AI that will be explained in the following unit: narrow and general AI.

Organizations like Gartner, McKinsey, or PricewaterhouseCoopers (PwC) predict a mind- blowing future of AI. Reports like the PWC report (2018) estimate that AI might make a contribution of 15.7 trillion USD to the global economy. Therefore, after discussing the two categories of AI, we will focus on the most important application areas of AI. Addi- tionally, we will explore how modern AI systems can be evaluated.

### Narrow versus General AI

Recent research topics in artificial intelligence distinguish between two types: artificial narrow intelligence (ANI), also referred to as weak artificial intelligence, and artificial general intelligence (AGI) or strong artificial intelligence. In ANI, systems are built to perform specialized functions in controlled environments whereas AGI comprises open-ended, flexible, and domain independent forms of intelligence like that which is expressed by human beings.

Even though many people believe that we already have some sort of strong artificial intelligence, current approaches are still implemented in a domain-specific way and lack the necessary flexibility to be considered AGI. However, there is a large consensus that it is only a matter of time until artificial intelligence will be able to outperform human intelligence. Results from a survey of 352 AI researchers indicate that there is a 50 percent chance that algorithms might reach that state by 2060 (Grace *et al*., 2017).

In the following, we will have a closer look at the underlying concepts of weak and strong artificial intelligence.

###### Artificial Narrow Intelligence

The term ANI or weak AI reflects the current and future artificial intelligence. Systems based on ANI can already solve complex problems or tasks faster than humans. How- ever, the capabilities of those systems are limited to the use cases for which they have been designed. In contrast to the human brain, narrow systems cannot generalize from a specific task to a task from another domain.

Modern AI Systems

For example, a particular device or system which can play chess, will probably not be able to play another strategy game like Go or Shogi without being explicitly program- med to learn that game. Voice assistants as Siri or Alexa can be seen as some sort of hybrid intelligences, which combine several weak AIs. Those tools are able to translate natural language and to analyze those words with their databases in order to complete different tasks. However, they are only able to solve a limited number of problems for which their algorithms are suitable and for which they have been trained for. For instance, currently, they would not be able to analyze pictures or optimize traffic.

In short, ANI includes the display of intelligence with regard to complex problem solv- ing and the display of intelligence relative to one single task.

###### Artificial General Intelligence

The reference point for which AGI is measured and judged against are the versatile cog- nitive abilities of humans. The goal of AGI is not only to imitate the interpretation of sensory input, but also to emulate the whole spectrum of human cognitive abilities. This includes all abilities currently represented by ANI, including the ability of domain- independent generalization. This means knowledge of one task can be applied to another in a different domain. This might also include motivation and volition. Some philosophical sources go one step further and require AGI to have some sort of con- sciousness or self-awareness (Searle, 1980). Developing an AGI would require the fol- lowing system capabilities:

* cognitive ability to function and learn in multiple domains
* intelligence on a human level across all domains
* independent ability to solve problems
* problem-solving abilities at an average human level over multiple domains
* ability to independently problem-solve
* abstract thinking abilities without drawing directly on past experience
* consideration of hypotheses without having previous experience
* perception of the whole environment in which the system acts
* self-motivation and self-awareness

Considering the current state of AGI, it is difficult to imagine developing a system that meets these requirements. In addition, both types of AI also entail the concept of superintelligence. This concept goes even further than current conceptions, and describes the idea that an intelligent system can reach a level of cognition that goes beyond human capabilities. This self-improvement might be achieved by a recursive cycle. However, this level of AI is above AGI and still very abstract.

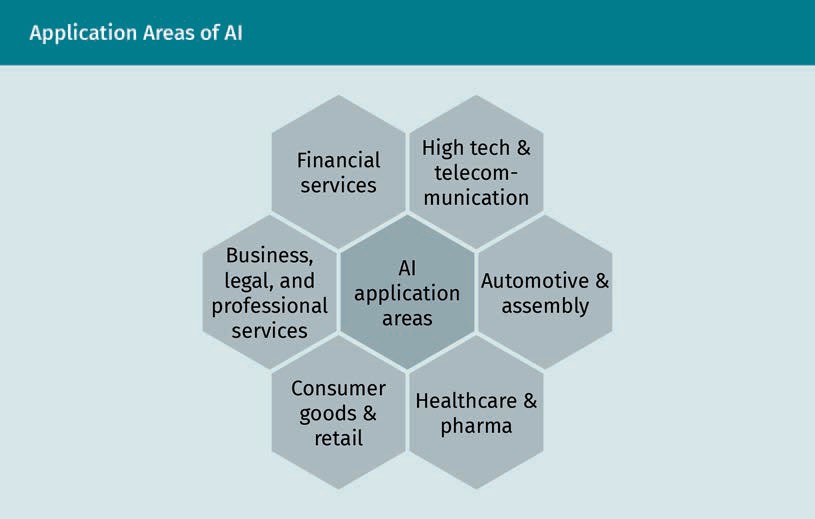
### Application Areas

Due to the latest advances in computational and data storage capabilities, in the past years, applications for AI have been continuously increasing. The options where AI can be applied are almost endless.

The growing interest is also corroborated by an increase in research activities. Accord- ing to the annual AI Index (Zhang *et al*., 2021), from 2019 to 2020, the number of journal publications on AI grew by 34.5 percent. Since 2010 AI papers increased more than twenty-fold. The most popular research topics have been natural language processing and computer vision which are important for various areas of application.

AI adoption The use of AI capa- bilities such as machine learning in at least one busi- ness function is called AI adoption.

In a global survey about the state of AI, McKinsey & Company (2021) identified the fol- lowing industries as the main fields of **AI adoption**: High Tech/Telecom, Automotive and Assembly, Financial Services, Business, Legal and Professional Services, Health- care/Pharma and Consumer Goods/Retail. In the following section, we will have a closer look at these fields.



The figure above summarizes the most important domains in which AI is used.

###### High Tech and Telecommunication

Due to the constant increase of global network traffic and network equipment, there has been a rapid growth of AI in telecommunication. In this area, AI can not only be used to optimize and automate networks but also to ensure that the networks, are healthy and secure.

Modern AI Systems

Using AI in predictive maintenance, it can help fix network issues even before they occur. Moreover, network anomalies can be accurately predicted when using self-opti- mizing networks.

Big data makes it possible to easily detect network anomalies and therefore prevent fraudulent behavior within them.

###### Automotive and Assembly

In the past years, autonomous driving has become a huge research topic. It will drasti- cally transform the automotive industry in the next decades from a steel-driven to a software-driven industry. Nowadays, cars are already equipped with many sensors to ensure the driver’s safety, for staying in–lane or emergency braking assistance.

Intelligent sensors can also detect technical problems based on the car or risks from the driver – such as fatigue or being under the influence of alcohol – and initiate appropriate actions.

Like in high tech and telecommunication, in assembly processes, AI can be used for predictive maintenance and to fix inefficiencies in the assembly line. Moreover, using computer vision, it is already possible to detect defects faster and more accurately than a human.

###### Financial Services

Financial services offer numerous applications for artificial intelligence. Intelligent algorithms enable financial institutions to detect and prevent fraudulent transactions and money laundering much earlier than was previously possible. Computer vision algorithms can be used to precisely identify counterfeit signatures by comparing them to scans of the originals stored in a database.

Additionally, many banks and brokers already use Robo-advising; Based on a user's investment profile, accurate recommendations about future investments can be made (D’Acunto *et al*., 2019). Portfolios can also be optimized based on AI applications.

###### Business, Legal, and Professional Services

Especially in industries where paperwork and repetitive tasks play an important role, AI can help to make processes faster and more efficient.

Significant elements of routine workflows are currently being automated using **robotic process automation** (RPA), which can drastically reduce administrative costs. Systems in RPA do not necessarily have to be enabled with intelligent AI capabilities. However, methods, such as natural language processing and computer vision, can help enhance those processes with more intelligent business logic.

Robotic process automation

The automated exe- cution of repetitive, manual, time con- suming or error prone tasks by soft- ware bots is descri- bed as robotic proc- ess automation.

The ongoing developments in big data technologies can help companies extract more information from their data. Predictive analytics can be used to identify current and future trends about the markets a company is in and react accordingly.

Another important use case is the reduction of risk and fraud, especially in legal, accounting, and consulting practices. Intelligent agents can help to identify potentially fraudulent patterns, which will allow for earlier responses.

###### Healthcare and Pharma

In the last few years, healthcare and pharma have been the fastest growing area adopt- ing AI.

AI-based systems can help detect diseases based on the symptoms. For instance, recent studies have been able to use AI–based systems to detect COVID–19 based on cough recordings (Laguarta *et al*., 2020).

Not only in diagnostics AI can offer many advantages. Intelligent agents can be used to monitor patients according to their needs. Moreover, regarding medication, AI can help find an optimal combination of prescriptions to avoid side effects.

Wearable devices – such as heart rate or body temperature trackers – can be used to constantly observe the vital parameters of a person. Based on this data, an agent can give advice about the wearer’s condition. Moreover, in case critical anomalies are detected, it is possible to initiate an emergency call.

###### Consumer Goods and Retail

The consumer goods and retail industry focuses on predicting customer behavior. Web- sites track how a customer’s profile changes based on their number of visits. This allows for personal purchase predictions for each customer. This data can not only be used to make personalized shopping recommendations but also to optimize the whole supply chain and direct about future research.

Market segmentation is, nowadays, no longer based on geographical regions such as province or country. Modern technologies allow it to segment customers’ behavior on a street-by-street basis. This information can be used to fine-tune operations and decide whether store locations should be kept or closed.

Additionally, the recent improvement in natural language processing technologies is increasingly used for chatbots and conversational interfaces. When it comes to cus- tomer retention and customer service, a well-developed artificial agent is key to ensur- ing customer satisfaction.

Modern AI Systems

###### Evaluation of AI Systems

As the above-mentioned examples illustrate, the application areas for modern AI sys- tems are almost unlimited. It is crucial that all data sets are independent from each other and follow a similar probability distribution.

To develop proper models for AI applications, the available data is split into three data sets:

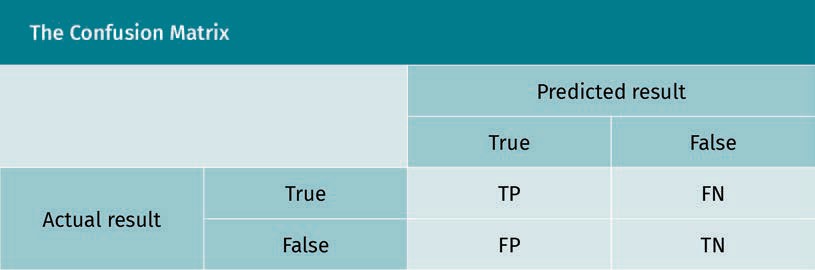
1. Training data set: As the name indicates, this data set is used to fit the parameters of an algorithm during the training process.
2. Development set: This data set is often also referred to as a validation set. It is used to evaluate the performance of the model developed using the training set and for further optimization. It is important that the development set contains data that have not been included in the training data.
3. Test set: Once the model is finalised using the training and the development set, the test set can be used for a final evaluation of the model. Like for the development set, it is important that the data in the test set has not been used before. The test set is only used once to validate the model and to ensure that it is not overfitted.

When developing and tuning algorithms, metrics should be in place to evaluate how well it performs independently and compared to other systems. In a binary classifica- tion task, accuracy, precision, recall, and F-score are metrics that are commonly used for this purpose.

For example, Financial services uses a binary classification task in fraud detection. A financial transaction can either be categorized as fraud or not. Based on this, we will have four categories of classification results:

1. True positives (TP): identifies samples that were correctly classified as positive, i.e. being fraudulent transactions
2. False positives (FP): all results that wrongly indicate a sample to be positive even though it’s negative, i.e., a non-fraudulent transaction being categorized as fraud
3. True negatives: marks classification results that were correctly classified as negative, i.e., non-fraudulent transactions that were also labeled as such
4. False negatives: classification results that were wrongly classified as negative even though they should have been positive, i.e., fraudulent transactions that were classi- fied as non-fraud

The classification results can be displayed in a confusion matrix, also known as error matrix. This is shown in the table below.



Using these categories, the above-mentioned metrics can be computed.

Accuracy is an indicator for how many samples were classified correctly. It can be com- puted as follows:

Accuracy = TP + TN

TP + TN + FP + FN

It measures which percentage of the total prediction was correct. Precision denotes the number of positive samples that were classified correctly in relation to all samples pre- dicted in this class:

Precision = TP

TP + FP

Recall indicates how many of the positively detected samples were identified correctly in relation to the total number of samples that should have been identified as such:

Recall = TP

TP + FN

Finally, the F-score combines precision and recall in one score:

F = 2 · precision · recall

precision + recall

In classification tasks with more than two classes, metrics can be calculated for every class. In the end the average of the values can be combined to one metric for all classes.

Even though the current application areas of AI are still using narrow AI algorithms, there are many possible use cases across several industries. More and more companies manage to support their business models with AI or to even create completely new ones.

Modern AI Systems

**Summary**

There are two types of AI: narrow and general. Current AI systems all belong to the category of ANI. ANI can solve complex problems faster than humans. However, its capabilities are limited to the domain for which it has been programmed. Even though the term ANI might suggest a limitation, it is embedded in many areas of our lives. In contrast to that, AGI (AI which has the cognitive abilities to transfer knowledge to other areas of application) remains a theoretical construct but is still an important research topic.

The application areas for AI are almost unlimited. AI has had a significant impact on today’s corporate landscape. Use cases, such as the optimization of service opera- tions, the enhancement of products based on AI, and automation of manual pro- cesses, can help companies towards optimizing their business functions. Those use cases stretch across a wide range of industries, be it automotive and assembly, financial services, healthcare and pharma, consumer goods, and many more.



# Unit 3

## Reinforcement Learning

#### STUDY GOALS

On completion of this unit, you will be able to …

… explain the basic principles of reinforcement learning.

… understand Markov decision processes.

… use the Q-learning algorithm.

DL-E-DLBDSEAIS01-U03

1. Reinforcement Learning

### Introduction

Imagine you are lost in a labyrinth and have to find your way out. As you are there for the first time, you do not know which way to choose to reach the door to leave. More- over, there are dangerous fields on the labyrinth and you should avoid stepping on them.

You will have four actions you can perform: move up, down, left, or right. As you do not know the labyrinth, the only way to find your way out is to see what happens when you perform random actions. Within the learning process, you will find out that there are fields on the labyrinth that will reward you by letting you escape the labyrinth. How- ever, there are also fields where you will receive a negative reward as they are danger- ous to step on. After some time, you will manage to find your way out without stepping on the dangerous fields from the experience you have made walking around. This proc- ess of learning by reward is called reinforcement learning.



In this unit, you will learn more about the basic ideas of reinforcement learning and the underlying principles. Moreover, you will get to know algorithms, such as Q-learn- ing, that can help you optimize the learning experience.

### What is Reinforcement Learning?

Generally, in machine learning, there exist three techniques to train a specific learning model: supervised, unsupervised, and reinforcement learning.

Reinforcement Learning

In supervised learning, a machine learns how to solve a problem based on a previously labeled data set. Typical application areas for supervised learning are regression and classification problems such as credit risk estimation or spam detection. Training those kinds of algorithms takes much effort because it requires a large amount of pre- labeled training data.

In unsupervised learning, training is performed using unlabeled data to discover the underlying patterns. Based on the input data, clusters are identified which can later be used for classification. This approach is often used to organize massive amounts of unstructured data such as customer behavior, to identify relevant peer groups.

Reinforcement learning techniques follow a more explorative approach. Algorithms based on this approach improve themselves by interacting with the environment. In contrast to supervised and unsupervised learning, there is no predefined data required. An agent learns on an unknown set of data based on the reward the environment returns to the agent. The following table summarizes the basic terms of reinforcement learning.

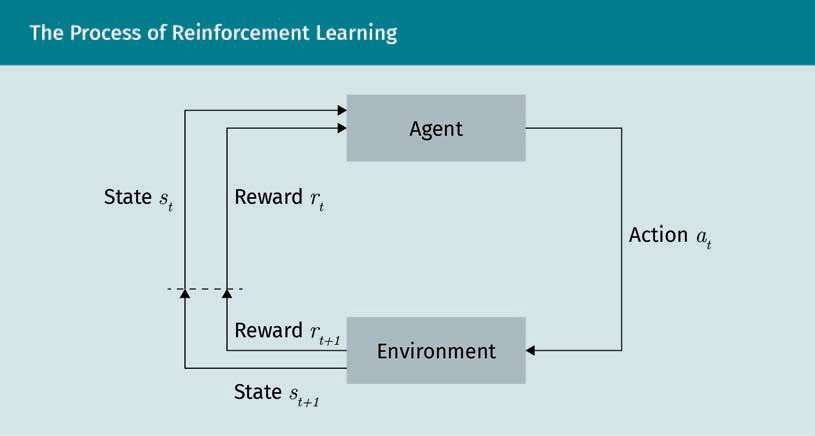
|  |  |
| --- | --- |
| Basic Terms of Reinforcement Learning | |
| Agent | Performs actions in an environment and receives reward for doing so |
| Action (A) | The set of all possible actions the agent can perform |
| Environment (E) | The scenario the agent must explore |
| State (S) | The current state of the agent in the given environment |
| Reward (R) | Immediate feedback from the environment to reward an agent's action |
| Policy (π) | The policy the agent applies to determine the next action based on the current state |
| Value (V ) | The long-term value of the current state S using the policy π |

Within the process of reinforcement learning, the agent starts in a certain state st ∈ S and applies an action at ∈ A st to the environment E, where A st is the set of actions available at state st. The environment reacts by returning a new state st + 1 and a reward rt + 1 to the agent. In the next step the agent will apply the next action at + 1 to the environment which will again return a new state and a reward.

In the introductory example, you are acting as the agent in the labyrinth environment. The actions you can perform are to move up, down, left, or right. After each move, you will reach another state by moving to another field in the labyrinth. Each time you per- form an action, you will receive a reward from the environment. It will be positive if you reach the door or negative if you step on a dangerous field. From your new position, the whole learning cycle will start again. Your goal will be to maximize your reward. The process of receiving a reward as a function of a state-action pair can be formalized as follows:

f st, at = rt + 1

The whole process of agent-environment interaction is illustrated in the figure below.



The process of an action being selected from a given state, transitioning to a new state, and receiving a reward happens repeatedly. For a sequence of discrete time steps t = 0, 1, 2, … starting at the state s0 ∈ S, the agent-environment interaction will lead to a sequence:

s0, a0, r1, s1, a1, r2, s2, a2, r3, s3, …

The goal of the agent is to maximize the reward it will receive during the learning proc- ess. The cycle will continue until the agent ends in a terminal state. The total reward R after a time T can be computed as the sum of rewards received at this point:

Rt = rt + 1 + rt + 2 + … + rT

This reward is also referred to as the Value V π s in the state s using the strategy π. In our example the maximum reward will be received once you reach the exit of the laby- rinth. We will have a closer look at the value function in the next section.

Reinforcement Learning

### Markov Decision Process and Value Function

To be able to evaluate different paths in the labyrinth, we need a suitable approach to compare interaction sequences. One method to formalize sequential decision-making is Markov Decision Processes (MDP). In the following, we will discuss how MDPs work.

###### The Markov Decision Process

MDPs are used to estimate the probability of a future event based on a sequence of possible events. If a present state holds all the relevant information about past actions, it is said to have the “Markov property”. In reinforcement learning, the Markov property is critical because all decisions and values are functions of the present state (Sutton & Barto, 2018), i.e., decisions are made depending on the environment’s state.

When a task in reinforcement learning satisfies the Markov property, this task is called a Markov decision process. If the Markov property is satisfied, in every state of a Markov chain, the probability that another state is reached depends solely on two factors: the transition probability of reaching the next state and the present state. The probability of reaching another state depends solely on the present state as well as the transition probability of reaching the next state. MDPs consist of the following components:

* + - States S
    - Actions A
    - Rewards for an action at a certain state ra = R s, a, s′
    - Transition probabilities for the actions to move from one state to the next state

Ta s, s′

Because of the Markov property, the transition function depends only on the current state:

P st + 1 st, at, st − 1, at − 1, … = P st + 1 st, at = Tat s, s′

Which action is picked in a certain state is described by the Policy π:

π s, a = p at = a st = s

Using our labyrinth example, the position at which you stand offers no information about the sequence of states you took to get there. However, your position in the laby- rinth represents all the required information for the decision about your next state, which means it has the Markov property.

###### The Value Function

In addition to the previously explained concepts, reinforcement learning algorithms use value functions. Value functions give an estimation about how good it is for an agent to be in that state and to perform a specific action in that given state (Sutton & Barto, 2018).

Previously, we learned that the value of a state can be computed as the sum of rewards received within the learning process. Additionally, a discount rate can be used to evalu- ate the rewards of future actions at the present state. The discount rate indicates the likelihood to reach a reward state in the future. This helps the agent select actions more precisely according to the expected reward. An action at + 1 will then be chosen to maximize the expected discounted return:

V π s =Eπ rt + 1 + γr t + 2 + … + γ T − 1 rT

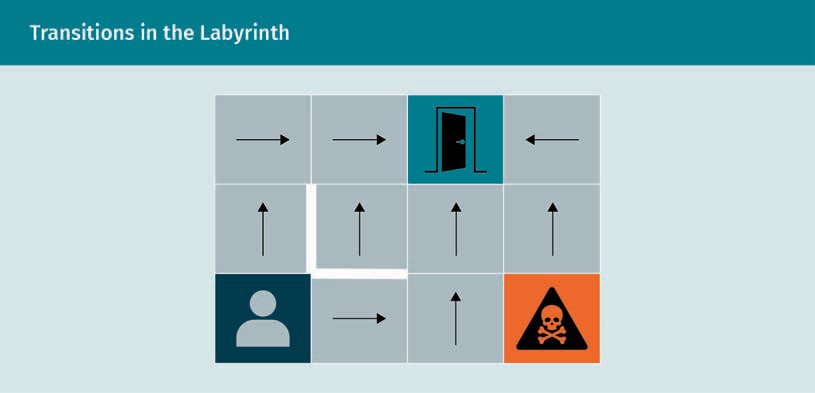
∞

=Eπ ∑ γkrt + k + 1 st = s

k = 0

where γ is the discount rate, with 0 ≤ γ ≤ 1, denoting the security of the expected return. A value of γ closer to 1 indicates a higher likelihood for future rewards. Espe- cially in scenarios where the length of time the process will take is not known in advance, it is important to set γ < 1, as otherwise the value function will not converge.

The following figure illustrates which action the agent should optimally perform in the respective states of the labyrinth to maximize the reward, i.e., trying to reach the exit and avoid the dangerous field.



Reinforcement Learning

### Temporal Difference and Q–Learning

So far, we discussed model-based reinforcement learning. That means that an agent tries to understand the model of the environment. All decisions are based on a value function. This value function is based on the current state and the future state where the agent will end.

In contrast to this, model-free approaches analyze the quality of an action to evaluate their actions. Q-Learning is a very well-known model-free reinforcement learning algo- rithm and is based on the concept of temporal difference learning. In the following, we will explain the underlying concepts of temporal difference and Q-learning.

###### Temporal Difference Learning

As temporal difference (TD) learning is a model-free approach, there is no model of the learning environment required. Instead, learning happens directly from the experience in a system that is partially unknown. As the name indicates, TD learning makes predic- tions based on the fact that there is often a correlation between subsequent predic- tions. The most prominent example to illustrate the principle of TD learning by Sutton (1988) is about forecasting the weather. Let’s say we want to predict the weather on a Monday. In a supervised learning approach, one would use the prediction of every day and compare it to the actual outcome. The model would be updated once it is Monday. In contrast to that, a TD approach compares the prediction of each day to the predic- tion of the following day, i.e., it considers the temporal difference between subsequent days and updates the predictions of one day based on the result of the previous day. Therefore, TD learning makes better use of the experience over time.

###### Q-Learning

One well-known algorithm based on TD learning is the Q-learning. After initialization, the agent will conduct random acts which are then evaluated. Based on the outcome of an action, the agent will adapt its behavior for the subsequent actions.

The goal of the Q-learning algorithm is to maximize the quality function Q s, a . The goal is to maximize the cumulative reward while being in a given state s by predicting the best action a (van Otterlo & Wiering, 2012). During the learning process Q s, a is iteratively updated using the **Bellman equation**:

Q s, a = r + γmaxa′Q s′, a′

All Q-values computed during the learning process are stored in the Q-Matrix. In every iteration, the matrix is used to find the best possible action. When the agent has to perform a new action, it will look for the maximum Q-value of the state-action pair.

Bellman equation The Bellman equa- tion computes the expected reward in an MDP of taking an action in a certain state. The reward is broken into the

immediate and the total future expected

reward.

The Q-learning Algorithm

In the following, we will itemize the Q-learning algorithm. The algorithm consists of an initialization and an iteration phase. In the initialization phase, all values in the Q-table are set to 0. In the iteration phase, the agent will perform the following steps:

1. Choose an action for the current state. In this phase there are two different strat- egies that can be followed:
   * Exploration: perform random actions in order to find out more information about the environment
   * Exploitation: perform actions based on the information which is already known about the environment based on the Q-table. The goal is to maximize the return
2. Perform the chosen action
3. Evaluate the outcome and get the value of the reward. Based on the result the Q- table will be updated.

**Summary**

Reinforcement learning deals with finding the best strategy for how an agent should behave in an environment to achieve a certain goal. The learning process of that agent happens based on a reward system which either rewards the agent for good decisions or punishes it for bad ones.

To model the process of the agent moving in the environment, Markov decision processes can be used. A value function can be applied to the system to better evaluate the quality of future decisions.

The Q-learning algorithm is a model-free approach from temporal difference learn- ing in which the agent gathers information about the environment based on explo- ration and exploitation.

Overall, the reinforcement learning process is very similar to learning through trial and error in real life.



# Unit 4

## Natural Language Processing

#### STUDY GOALS

On completion of this unit, you will be able to…

… explain the historical background of NLP.

… name the most important areas of application.

… distinguish between statistical- and rule-based NLP techniques.

… understand how to vectorize data.

DL-E-DLBDSEAIS01-U04

1. Natural Language Processing

### Introduction

Natural language processing (NLP) is one of the major application domains in artificial intelligence.

NLP can be divided into three subdomains: speech recognition, language understand- ing, and language generation. Each will be addressed in the following sections. After an introduction to NLP and its application areas, you will learn more about the basic NLP techniques and how data vectorization works.

### Introduction to NLP and Application Areas

NLP is an interdisciplinary field with roots in computer science (especially the area of artificial intelligence), cognitive science, and linguistics. It deals with processing, under- standing, and generating natural language (Kaddari *et al*., 2021). In human-computer interaction, NLP has a key role when it comes to making the interaction more natural. Therefore, the goal of NLP is to use and interpret language on a similar level to that of humans. This does more than just help humans to interact with the computer using natural language; there are many interesting use cases, ranging from automatic machine translation to generating text excerpts, or even complete literature works. As mentioned above, there are three subdomains in NLP:

1. Speech recognition: identifies words in spoken language and includes speech-to- text processing
2. Natural language understanding: extracts the meaning of words and sentences as well as reading comprehension
3. Natural language generation: is the ability to generate meaningful sentences and texts.

All these subdomains build on methods from artificial intelligence and form the basis for the areas of application of NLP.

###### Historical Developments

Early NLP research dates back to the seventeenth century, when Descartes and Leibnitz conducted some early theoretical research about NLP (Schwartz, 2019). It became a technical discipline in the mid-1950s. The geopolitical tension between the former Soviet Union and the United States led to an increased demand for English-Russian translators. Therefore, it was attempted to outsource translation to machines. Even though the first results were promising, machine translation turned out to be much more complex than originally thought, especially as no significant progress could be

Natural Language Processing

seen. In 1964 the Automatic Language Processing Advisory Committee classified the NLP technology as “hopeless” and decided to temporarily stop the research funding in this area. This was seen as the start of the NLP winter.

Almost 20 years after the NLP winter began, NLP started to regain interest. This was due to the following three developments:

1. Increase of computing power: Computing power significantly increased, allowing for more computationally intensive algorithms following Moore’s law.
2. Shift of paradigms: Early language models were based on a grammatical approach that tried to implement complex rule-based systems to deal with the complexity of everyday language. More recent research had shifted towards models that are based on statistical and decision-theoretic foundations, such as decision trees.
3. Part-of-speech-tagging (POS): For this technique, a text is split into smaller units, i.e., individual sentences, words, or sub-words. Using POS tagging, grammatical word functions and categories are added to a given text. This allows to describe speech using **Markov models**. In contrast to approaches that consider the whole history, this is a major reduction of complexity.

Taken together, the shift to statistical, decision-theory, and machine learning models increased the robustness of NLP, especially concerning their ability to deal with unknown constellations. Moreover, the improved computing power allowed to process a much bigger amount of training data which was now available because of the growing amount of electronic literature. This opened up big opportunities for the available algorithms to learn and improve.

###### NLP and the Turing Test

One of the early pioneers in AI was the mathematician and computer scientist Alan Mathison Turing. In his research, he formed the theoretical foundation of what became the Turing test (Turing, 1950). In the test, a human test person uses a chat to interview two chat partners: another human and a chatbot. Both try to convince the test person that they are human. If the test person cannot identify which of their conversational partners is human and which is the machine, the test is successfully passed. According to Turing passing the test allows the assumption that the intellectual abilities of a com- puter are at the same level as the human brain.

The Turing test primarily addresses the natural language processing abilities of a machine. Therefore, the Turing test has often been criticized as being too focused on functionality and not on consciousness. One early attempt to pass the Turing test was done by Joseph Weizenbaum who developed a computer program to simulate a conver- sation with a psychotherapist (Weizenbaum, 1966). His computer program ELIZA was one of the first conversational AIs. To process the sentence entered by the user, ELIZA utilizes rule-based pattern matching combined with a thesaurus. The publication got some remarkable feedback from the community. Nevertheless, the simplicity of this approach was soon recognized and according to the expectations from the community, ELIZA did not pass the Turing test.

Markov models

In a Markov model, the next state is defined based on the current state and a set of transition probabilities.

In 2014 the Chatbot “Eugene Goostman” was the very first AI which seemed to have passed the Turing test. The Chatbot pretended to be a 13-year-old boy from Ukraine who was not a native English speaker. This trick was used to explain that the bot did not know everything and sometimes made mistakes with the language. However, this trick was also the reason why the validity of the experiment was later questioned (Mas- nick, 2014).

###### Application areas of NLP

Now we will briefly describe the major application areas of NLP.

Topic identification

As the name indicates, topic identification deals with the challenge to automatically find the topics of a given text (May *et al*., 2015). This can either be done in a supervised or in an unsupervised way. In supervised topic identification, a model can, for instance, be trained on newspaper articles that have been labeled with topics, such as politics, sports, or culture. In an unsupervised setting, the topics are not known in advance. In this case, the algorithm has to deal with topic modeling or topic discovery to find clus- ters with similar topics.

Popular use cases for topic identification are, for instance, social media and brand monitoring, customer support, and market research. Topic identification can help find out what people think about a brand or a product. Social media provides a tremendous amount of text data that can be analyzed for these uses cases. Customers can be grou- ped according to their interests, and reactions to certain advertisements or marketing campaigns can be easily analyzed. When it comes to market research, topic identifica- tion can help when analyzing open answers in questionnaires. If those answers are pre-classified, it can reduce the effort to analyze open answers.

Moreover, in customer support, topic identification can be beneficial by categorizing the customers’ requests by topics. Automatically forwarding requests to specialized employees can not only reduce costs, but also increase customer satisfaction.

Text summarization

Text summarization deals with methods to automatically generate summaries of a given text that contain the most relevant information from the source. Algorithms for text summarization are based on extractive and abstractive techniques. Extractive algo- rithms produce a summary of a given text by extracting the most important word sequences. Abstract techniques, conversely, generate summaries by creating a new next and rewriting the content of the original document.

A common text summarization technique that works in an unsupervised extractive way is TextRank (Mihalcea & Tarau, 2004). This algorithm compares every sentence of a given text with all other sentences. This is done by computing a similarity score for every pair of sentences. A score closer to one indicates a higher similarity between one sentence and another sentence that represents the content in a good way. For each sentence, the scores are summarized to get a sentence rank. After sorting the senten-

Natural Language Processing

ces according to their rank, it is easy to evaluate the importance of each one and create a summary from a predefined number with the highest rank. There are two major chal- lenges when dealing with supervised extractive text summarization, as training requires a lot of hand-annotated text data. These are:

1. It is necessary that the annotations contain the words that have to be in the sum- mary. When humans summarize texts, they tend to do this in an abstract way. There- fore, it is hard to find training data in the required format.
2. The decision about what information should be included in the summary is subjec- tive and depends on the focus of a task. While a product description would focus more on the technical aspects of a text, a summary of the business value of a prod- uct will put the emphasis on completely different aspects.

A typical use case for text summarization is presenting a user a preview of the content of search results or articles. This makes it easier to quickly analyze a huge amount of information. Moreover, in question answering, text summarization techniques can be used to help a user find answers to certain questions in a document.

Sentiment analysis

Sentiment analysis captures subjective aspects of texts (Nasukawa & Yi, 2003), such as analyzing the author’s mood on a tweet on Twitter. Like topic identification, sentiment analysis deals with text classification. The major difference between topic identification and text classification is that topic identification focuses on objective aspects of the text while sentiment analysis centers on subjective characteristics like moods and emotions.

The application areas for sentiment analysis are manifold. Customer sentiment analysis has gained much traction as a research field lately. The ability to track customers’ sen- timents over time can, for instance, give important insights about how customers react to changes of a product/a service or how external factors like global crises influence customers’ perceptions. Social networks, such as Facebook, Instagram, and Twitter, pro- vide huge amounts of data about how customers feel about a product. Having a better understanding of customer’s needs can help modify and optimize business processes accordingly. Detecting emotions from user-generated content comes with some big challenges when dealing with irony/sarcasm, negation, and multipolarity.

There is much sarcasm in user-generated content, especially in social media. Even for humans, it can sometimes be hard to detect sarcasm, which makes it even more diffi- cult for a machine. Let us, for instance, look at the sentence

“Wow, your phone has an internal storage of 1 Gigabyte?”

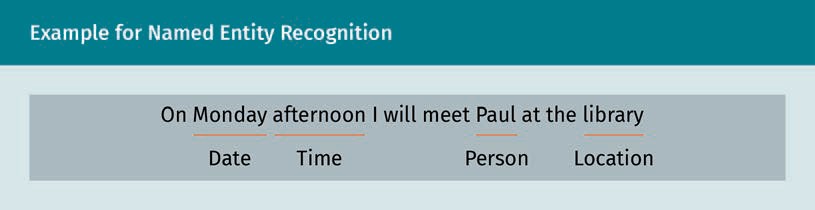
Only a few years back this would have been a straightforward sentence. Now, if said about a modern smartphone, it is easy for a human to tell that this statement is sar- castic. While there has been some recent success in sarcasm detection using methods from deep learning (Ghosh & Veale, 2016), dealing with sarcasm remains a challenging task.

Negation is another challenge when trying to detect a statement's sentiment. Negation can be explicit or implicit, and also comes with the morphology of a word denoted by prefixes, such as “dis-“ and “non-,“ or suffixes, such as “-less”. Double negation is another language construct that can be easily misunderstood. While most of the time double negatives will cancel each other, in some contexts it can also intensify the neg- ation. Considering negation in the model used for sentiment analysis can help to sig- nificantly increase the accuracy (Sharif *et al*., 2016).

An additional challenge in sentiment analysis can be multipolarity, meaning that some parts of the text can be positive while others are negative. Given the sentence “The dis- play of my new phone is awesome, but the audio quality is really poor,” the sentiment for the display is positive while it is negative for the speakers. Simply calculating the average of the sentiment might lead to information loss. Therefore, a better approach to tackle this issue would be to split the sentence into two parts: one for the positive review of the display and one for the negative feedback about the speakers.

Named entity recognition

Named entity recognition (NER) deals with the challenge of locating and classifying named entities in an unstructured text. Those entities can then be assigned to catego- ries such as names, locations, time and date expressions, organizations, quantities, and many more. NER plays an important role in understanding the content of a text. Espe- cially for text analysis and data organization, NER is a good starting point for further analysis. The following figure shows an example of how entities can be identified from a sentence.



NER can be used in all domains where categorizing text can be advantageous. For instance, tickets in customer support can be categorized according to their topics. Tick- ets can then automatically be forwarded to a specialist. Also, if data has to be anony- mized due to privacy regulations, NER can help to save costs. It can identify personal data and automatically remove it. Depending on the quality of the underlying data, manual cleanup is no longer necessary. Another use case is to extract information from candidate resumes in the application process. It can significantly decrease the work- load of the HR department, especially when there are many applicants (Zimmermann *et al*., 2016).

The biggest challenge in NER is that to train a model, it is necessary to have a large amount of annotated data for training available. The model will later always focus on the specific tasks/the specific subset of entities on which it has been trained.

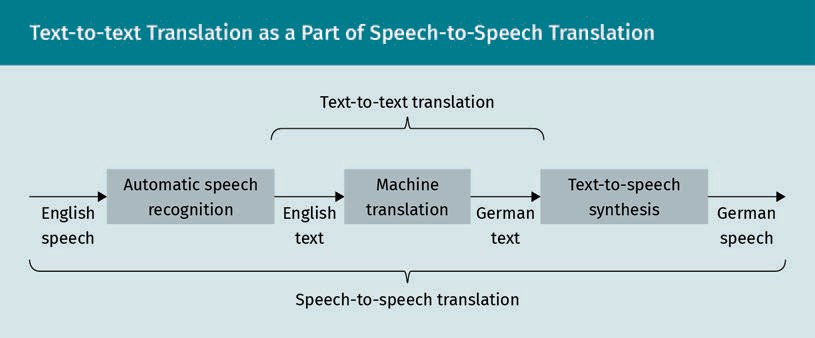
Natural Language Processing

Translation

Machine translation (MT) is a subfield of NLP that combines several disciplines. Using methods from artificial intelligence, computer science, information theory, and statis- tics, in MT text or speech are automatically translated from one language to another.

In the last decades, the quality of MT has significantly improved. In most cases, the quality of machine translations is still not as good as those done by professional trans- lators. However, combining MT and manual post-processing is nowadays often faster than translating everything manually. Like in any other area of NLP, the output quality depends significantly on the quality of the training data. Therefore, often domain-spe- cific data are used. While in the past, the most commonly used method was **statistical machine translation** (SMT), **neural machine translation** (NMT) has become more popu- lar (Koehn & Knowles, 2017).

MT can be used for text-to-text translations as well as speech-to-speech translations. Using MT for text can help quickly translate text documents or websites, assist profes- sional translators to accelerate the translation process, or as a part of a speech-to- speech translation system. As globalization progresses, MT has become more important every day. In 2016, Google was translating over 100 billion words per day in more than 100 languages. (Turovsky, 2016) The following figure shows how text-to-text translation is interlinked with speech-to-speech translation.



Text-to-text translation and speech-to-speech translation are becoming increasingly important. This process has been accelerated by the huge increase of video chats and conferences in recent years. Speech-to-speech translation can help bridge language barriers using applications such as the Skype translator.

As the figure illustrates, the core of a speech-to-speech translation system is a text-to- text translation system. Before the translation starts, speech has to be converted into text using methods from automatic speech recognition (ASR). After the translation, text- to-speech (TTS) synthesis is used to produce speech from the translated text. There- fore, the quality of the output does not only depend on the quality of the MT compo- nent, but also on the quality of the ASR and TTS components, which makes speech-to- speech-translation challenging.

Statistical machine translation

In statistical machine translation, translations are gen- erated using statisti- cal models that were built based on the analysis of bilingual text corpora.

Neural machine translation

In neural machine translation, an artifi- cial neural network is used to learn a statistical model for MT.

Nowadays, the two biggest challenges in MT are domain mismatch and under- resourced languages. Domain mismatch means that words and sentences can have dif- ferent translations based on the domain. Thus, it is important to use domain adaption when developing an MT system for a special use case.

For some combinations of languages in MT, there are no bilingual text corpora available for source and target language. One approach to solving the problem of under- resourced languages is to use pivot MT. In pivot MT, the source and target language are bridged using a third language (Kim *et al*., 2019). When, for instance, translating from Khmer (Cambodia) to Zulu (South Africa), a text will first be translated from Khmer to English and afterwards from English to Zulu.

Chatbots

Chatbots are text-based dialogue systems. They allow interaction with a computer based on text in natural language. Based on the input, the system will reply in natural language. Sometimes, chatbots are used in combination with an avatar simulating a character or personality. One of the most popular chatbots was ELIZA imitating a psy- chotherapist. Chatbots are often used in messenger apps, such as Facebook, or website chats. Moreover, they form the basis for digital assistants, like Alexa, Siri, or Google Assistant.

Chatbots can be categorized according to their level of intelligence:

* Notification assistants (level 1): These chatbots only interact unidirectionally with the user. They can be used for notifications about events or updates (i.e., push noti- fications).
* Frequently asked questions assistants (level 2): Those bots can bi-directionally interact with a user. They can interpret the user’s query and find an appropriate answer in a knowledge base.
* Contextual assistants (level 3): these chatbots can not only interact bidirectionally, but also be context-aware based on the conversation history.

In the future, it is likely that further levels of chatbots will evolve. A chatbot is based on three components:

1. Natural language understanding (NLU): This component parses the input text and identifies the intent and the entities of the user (user information).
2. Dialogue management component: The goal of this component is to interpret the intent and entities identified by the NLU in context with the conversation and decide the reaction of the bot.
3. Message generator component: Based on the output of the other components, the task of this component is to generate the answer of the chatbot by either filling a predefined template or by generating a free text.

Chatbots can save a lot of time and money. Therefore, use cases are increasing contin- uously. They are normally available 24/7 at comparatively low costs and can easily be scaled if necessary. In customer service, they can not only reply to customers’ requests,

Natural Language Processing

but also give product recommendations or make travel arrangements, such as hotel or flight reservations. If a request is too complicated for a bot, there are usually interfaces to forward requests to a human support team.

In marketing, chatbots can be used to generate leads. An increasing number of com- pany websites are already using bots to ask sales-oriented questions to otherwise anonymous visitors. The replies can be used to acquire new leads. Moreover, the chat- bot can provide guidance or present new products or information about special offers. Using chatbots as personal sales agents can also help reduce the efforts of humans, which allows them to focus on more complex tasks.

### Basic NLP Techniques

Early systems for NLP applied rules based on linguistic structures. Those rules were often handwritten and only applied to the domain for which they were designed. Nowadays, most NLP systems use statistical methods from machine learning.

###### Rule-Based Techniques

Rule-based techniques for NLP use a set of predefined rules to tackle a given problem. Those rules try to reproduce the way humans build sentences. A simple example for a rule-based system is the extraction of single words from a text based on the very sim- ple rule “divide the text at every blank space”. Looking at terms like “New York” already illustrates how fast a seemingly simple problem can get complicated. Therefore, more complex systems are based on linguistic structures using formal grammars.

The rule-based approach implies that, to build a system, humans have to be involved in the process. Because of this, one of the major advantages of rule-based systems is the explainability: As the rules have been designed by humans, it is easy to understand how a task has been processed and to locate errors.

Rule-based systems can be developed flexibly. Typically, it is unnecessary to make changes to the core of an application when rules are changed, be it by adding new rules or correcting existing ones. Another advantage of rule-based systems is that the amount of training data required to develop the system is comparatively small.

The major drawback of the rule-based approach is that it requires experts to build a set of appropriate rules. Moreover, rule-based systems are normally built in a domain- specific way, which makes it difficult to use a system in a domain for which it was not designed.

###### Statistical-Based Techniques

Since computational power has increased in the past decades, systems based on stat- istical methods – which are often subsumed under the term machine learning – have replaced most of the early rule-based systems. Those methods follow a data-driven approach. The models generated by statistical-based methods are trained with a huge amount of training data to derive the rules of a given task. After that, the models can be used to classify a set of unknown data to make predictions. In contrast to rule- based systems, statistical-based systems do not require expert knowledge about the domain. They can easily be developed based on existing methods and improved by providing appropriate data. Also transferring the model to another domain is much easier than for rule-based systems.

However, one disadvantage of systems based on statistical machine learning techni- ques is that much annotated training data is required to produce good results, whereas rule-based systems can perform well when there is only limited data available for a specific task.

###### Tasks

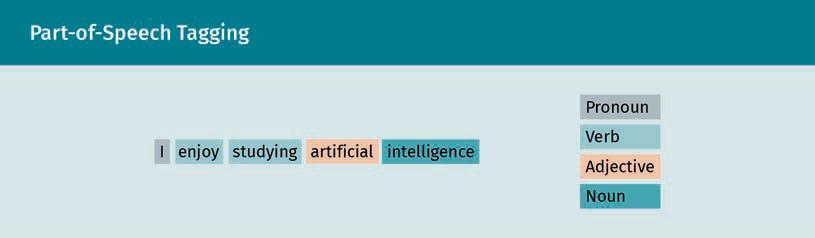
In general, NLP tasks can be divided into four categories, syntax, semantics, discourse, and speech. In the following, we will give an overview of those tasks.

Syntax

Syntactical tasks in NLP deal with the features of language such as categories, word boundaries, and grammatical functions. Typical tasks dealing with the syntax are toke- nization and part-of-speech (POS) tagging.

The goal of tokenization is to split a text into individual units such as words, sentences, or sub-word units. For instance, the sentence “I enjoy studying artificial intelligence.” could be tokenized into “I” “enjoy” “studying” “artificial” “intelligence” “.” .

POS tagging – also called grammatical tagging – goes one step further and adds gram- matical word functions and categories to the text. The following example illustrates how a sentence can be analyzed using POS tagging.



Natural Language Processing

Syntactic ambiguity, i.e., words which cannot be clearly assigned to a category, are a big challenge in NLP.

One commonly used example for syntactic ambiguity is the sentence

“Time flies like an arrow”

which can be interpreted in many different ways. Two of the possible interpretations are

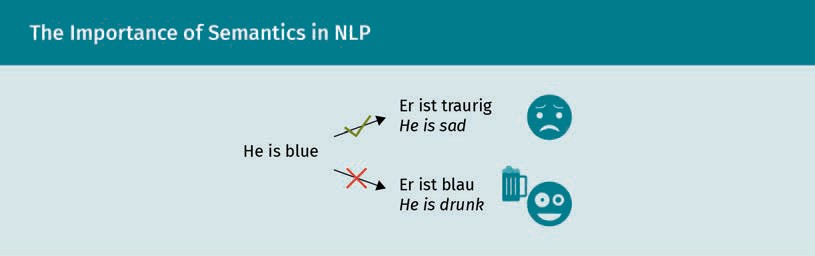
1. Time passes as quickly as an arrow
2. There exists a particular arrow such that every “time fly” (insect) likes that arrow

In the first interpretation, “like” is a comparative preposition, while in the second inter- pretation it is a verb.

Semantics

The focus of semantic tasks is on the meaning of words and sentences. Understanding the meaning of a text is essential for most application areas of NLP. In sentiment analy- sis, subjective aspects of the language are analyzed. For instance, when analyzing posts on social media, it is important to understand what the text means to identify whether it is positive or negative. Named entity recognition (NER) is another research field where semantics are important for correct classification results. Identifying entities, such as names, locations, or dates, from a given text cannot be done without under- standing the semantics of a text. In topic identification, a given text is labeled with a topic. Therefore, it is important to understand what the text is about. For instance, newspaper articles could be labeled with topics such as “politics”, “culture”, or “weather”.

If NLP is used for answering questions, a computer needs to create an appropriate answer to a certain question. Assuming a question answering algorithm was trained on this course book, the algorithm might display this section when asked “What are the typical tasks in NLP?” For this purpose, the semantics of this section must be interpre- ted correctly. Also, in machine translation, understanding the correct meaning of a text is essential. Otherwise, the translation will yield results that are hard to understand or even wrong.



The figure above illustrates how important it is to properly understand the semantics of a text.

Discourse

Discourse deals with text that is longer than a single sentence. It is important for tasks, such as topic identification and text summarization, where an algorithm produces a summary of a given text by extracting the most important sentences. Analyzing the dis- course of a text involves several sub-tasks, like identifying the topic structure, analysis of the coreference, and the conversation structure.

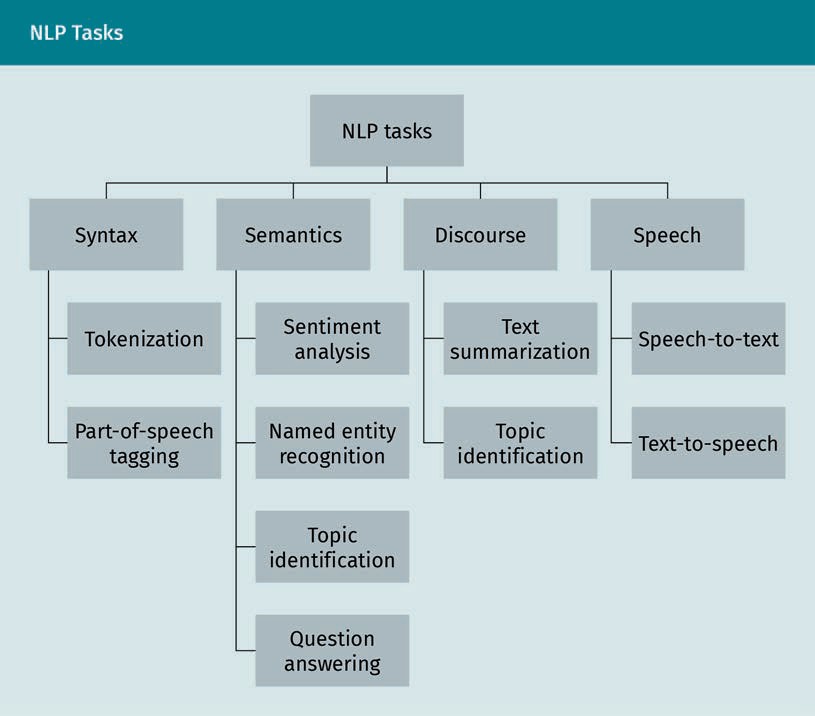
Speech

Speech tasks in NLP are all about spoken language. In speech tasks two sub-tasks can be distinguished:

* 1. Speech-to-text (STT): Also referred to as automatic speech recognition (ASR), it con- verts spoken language into text.
  2. Text-to-speech (TTS), or speech synthesis, which deals with transforming a written text into spoken language.

Both are important for conversational interfaces, such as voice assistance, like Siri or Alexa. The following figure summarizes the typical tasks in NLP.

Natural Language Processing



### Vectorizing Data

In machine learning, algorithms only accept numeric input. Therefore, if we want to extract information from an unstructured text, we need to find a way that the computer can process it. For this purpose, the text has to be converted into a numerical format, which the computer can process.

In the following, we want to introduce two approaches to how words can be embedded into a semantic vector space: the bag-of-words approach, which is simple, and the more powerful concept of neural word and sentence vectors.

###### Bag-of-Words

One of the easiest approaches to convert textual information into numbers is the bag- of-words (BoW) model. Using BoW a text is represented by a vector that describes the number of word occurrences in a given text document. The term “bag” refers to the fact

that once the words are put into the unique set of words describing a text, all informa- tion about the structure or order of the words in a text is discarded. To understand the BoW approach, we will use the following example text:

* + - Darren loves dogs
    - Darren does not like cats
    - Cats are not like dogs

In the first step, we need to identify all unique words from the text. For this purpose, we use tokenization.

In the above text, the following words are used:

Darren, loves, dogs, does, not, like, cats, are

In the next step, we need to score the words in every sentence. As we know that our vocabulary consists of 8 words, the resulting vector will have a length of 8. The BoW vectors for the sentences above will look as follows:

* + - [1, 1, 1, 0, 0, 0, 0, 0]
    - [1, 0, 0, 1, 1, 1, 1, 0]
    - [0, 0, 1, 0, 1, 1, 1, 1]

There are different methods to score the words in the BoW model. In the above senten- ces every word only occurred once, therefore the resulting vectors are a binary repre- sentation of the text. If the whole text from the above example were summarized in one vector, the following options are available:

* + - Boolean representation: the vector simply indicates if a word occurs or not [1, 1, 1, 1, 1, 1, 1, 1].
    - Count of words: the resulting vector reflects how often a word occurs [2,1,2,1,2,2,2,1].

As you will notice, this representation no longer contains any information about the original order of the words.

Limitations of Bag-of-Words

Taken together, the BoW model is simple, which induces some major disadvantages:

Natural Language Processing

* Selection of vocabulary: The vocabulary of the model has to be selected very care- fully. The balance between the size of the model and sparsity must always be kept in mind. The larger the vocabulary, the higher will be the sparsity of the vectors.
* Risk of high sparsity: For computational reasons, it is more difficult to model sparse representation of data, as the complexity of time and space will increase with higher sparsity. Moreover, it is more difficult to make use of the data if only a little informa- tion is contained in a large representational space.
* Loss of meaning: Using BoW, neither word order nor context nor sense are consid- ered. In our example, the different meanings of “like” (once being used as a preposi- tion and once as a verb) gets completely lost. In situations like that, the BoW model does not perform well.

###### Word Vectors

To be able to embed words in a semantic vector space they can be represented as word vectors. Linear operations can be applied to find word analogies and similarities. These word similarities can, for instance, be based on the cosine similarity. Most importantly, once words are transformed into word vectors, they can be used as an input for machine learning models, like artificial neural networks and linear classifiers. In the following, three vectorization methods will be presented: Word2Vec, TD-IDF and GloVe.

Word2Vec

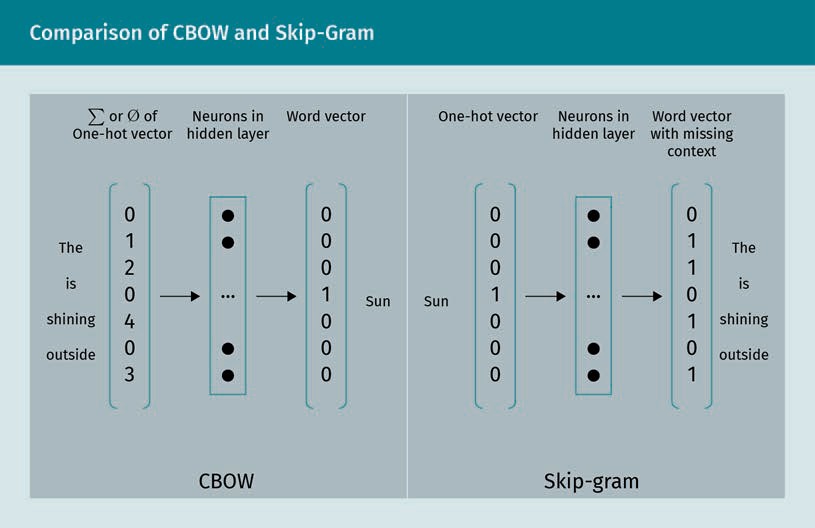
The Word2Vec model is based on a simple neural network. The neural network gener- ates word embeddings based on only one hidden layer (Mikolov *et al*., 2013). A research milestone was passed when Google Research published the model in 2013. The input layer of the neural network expects a “one-hot vector.” The one-hot vector is a BoW vector for one single word. This means that all indices of that vector are set to 0 except for the index of the word, which is analyzed. This index is set to 1.

Training the neural network for Word2Vec requires a large text corpus. This could, for instance, be a Wikipedia dump. When the training is performed, a fixed-length word window with length N is slid over the corpus. Typical values for N would, for example, be N = 5 or N = 10.

In Word2Vec there are two prediction models:

1. Continuous Bag-of-Words (CBOW): This model can be used if the goal is to predict one missing word in a fixed window in the context of the other N − 1 words. As an input vector, we can either use the average or the sum of the one-hot vector.
2. Skip-gram: If we have one world within a fixed window, with this model we can pre- dict the remaining N − 1 context words.

The difference between both models is illustrated in the figure below.



One important aspect of CBOW is that the prediction outcome is not influenced by the order of the context words. In skip-gram, nearby context words are weighted more heavily than more distant context words. While CBOW generally performs faster, the skip-gram architecture is better suited for infrequent words.

When training Word2Vec, the goal is to maximize the probabilities for those words that appeared in the fixed window of the analyzed sample from the data corpus used for training. The function we receive from this process is the objective function.

In an NLP task, the goal is usually not to find a model to predict the next word based on a given text snippet, but to analyze the syntax and semantics of a given word or text. If we remove the output layer of the model we generated before and look at the hidden layer instead, we can extract the output vector from this layer. Neural networks usually develop a strong abstraction and generalization ability on their last layers. It is, there- fore, possible to use the output vector of the hidden layer as an abstract representa- tion of the features from the input data. Thus, we can use it as an embedding vector for the word we want to analyze.

Nowadays, there are many pre-trained Word2Vec models available for various lan- guages that can easily be adapted for specific NLP tasks.

Term frequency-inverse document frequency

In BoW, the frequency of vocabulary words does only reflect words that are contained in the document. Therefore, all words are given the same weight when analyzing a text, no matter their importance. Term frequency-inverse document frequency (TF-IDF) is a statistical measure from information retrieval that tackles this problem and is one of the most commonly used weighting schemes in information retrieval (Beel *et al*., 2016). In TF-IDF the term frequency (TF) is combined with the inverse document frequency

Natural Language Processing

(IDF). The relevance of a word increases with its frequency in a certain text but is com- pensated by the word frequency in the whole data set. For the computation of TF-IDF we need the following parameters:

* Term frequency (TF) reflects how often a term t occurs in a document d. The word order is not relevant in this case. The number of occurrences is weighted by the total number of terms in the document:

TF t, d = number of occurences of t in d

number of words in d

* Document frequency (DF) indicates the percentage of documents including a spe- cific term t in relation to the total number of documents D. This can be seen as an indicator for the importance of the text.

DF t, d, D = number of documents d containing t

total number of documents D

* Inverse document frequency (IDF) tests the relevance of a particular term. As the name suggests, it is the inverse of the document frequency logarithmically scaled:

IDF t = log 1

DF t, d, D

The final TF-IDF score for a term can be computed as follows:

TFIDF t, d = TF t, d · IDF t

High values of TF-IDF indicate words that occur often in a document while the number of documents that contain the respective term is small compared to the total amount of documents. Therefore, TF-IDF can help find terms in a document that are most important in a text.

GloVe

Global Vectors for word representation (GloVe) is another vectorization method com- monly used in NLP. While Word2Vec is a predictive model, GloVe is an unsupervised approach based on the counts of words. It was developed because Pennington *et al*. (2014) concluded that the skip-gram approach in Word2Vec does not fully consider the statistical information when it comes to word co-occurrences. Therefore, they com- bined the skip-gram approach with the benefits of **matrix factorization**. The GloVe model uses a co-occurrence matrix, which contains information about the word con- text. The developed model has been shown to outperform related models, especially for named entity recognition and similarity tasks (Pennington *et al*., 2014).

Matrix factorization This is used to reduce a matrix into its components to simplify complex matrix operations.

###### Sentence Vectors

So far, we have learned how to represent words as vectors. However, various NLP tasks, like question answering or sentiment analysis, require not only the analysis of a single word but of a whole sentence or paragraph. Therefore, we also need a way how to encode a sequence of words to be able to process it with a learning algorithm.

One approach is to build an average of the vectors of a sentence from Word2Vec and use the resulting vectors as input for a model. However, this method would come with the disadvantage that the word order is no longer included in the word encodings. For instance, the sentences “I put the coffee in the cup” and “I put the cup in the coffee” contain the same words. Only the word order makes the difference in the sentence.

To tackle the problem of dealing with text snippets of various lengths, there exist sev- eral approaches. In the following sections, we will present a selection of those algo- rithms. Please note that in the following the term “sentence” will also be used to repre- sent a whole paragraph of text, not only as a sentence in a strict grammatical way.

Skip-thought

In the skip-thought vectors approach (Kiros *et al*., 2015) the concept of the skip-gram architecture we introduced previously in the section about the Word2Vec approach is transferred to the level of sentences.

Like Word2Vec, skip-thought requires a large text corpus to train the model. In contrast to Word2Vec, instead of using a sliding word window, skip-thought analyzes a triple of three consecutive sentences. The resulting model is a typical example of an encoder- decoder architecture. The middle sentence from the triple is used as an input for the encoder. The encoder produces an output, which is connected to the decoder. There are two ways to optimize the model: the decoder can either be used to predict the follow- ing or the previous sentence of the sentence the encoder received.

There are some NLP tasks that do not require a prediction model. For those tasks, the decoder part is no longer needed after the training and can be discarded. To get the vector representation of the sentence, we can use the output vector of the encoder.

In case we use the model to only predict the following or the previous sentence, the result is a uni-skip vector. When concatenating two uni-skip vectors so that one pre- dicts the previous and the other predicts the next sentence, the result is called a bi- skip vector. If n-dimensional uni-skip vectors are combined with n-dimensional bi-skip vectors, the result will be a 2n-dimensional combine-skip vector. In a comparison of several combine-thought models, the combine-skip model has been proven to perform slightly better.

There is a pre-trained English language model available to the public based on the BookCorpus dataset.

Natural Language Processing

Universal sentence encoder (USE)

The universal sentence encoder (USE) is a family of models for sentence embedding that was developed by Google Research (Cer *et al*., 2018). There are two architecture variants of the USE. One variant uses a deep averaging network (DAN); (Iyyer *et al*., 2015) and is faster but less accurate, while the other variant utilizes a transformer model.

Also for USE, there are pre-trained models available to the public: one English model and one multilingual model (Chidambaram *et al*., 2019). These models are both based on the DAN architecture.

Bidirectional encoder representations from transformers (BERT)

As the name indicates, BERT (Devlin *et al*., 2018) was based on the transformer architec- ture. Like USE this model was introduced by Google Research. The language model is available open-source and has been pre-trained on a large text corpus in two com- bined and unsupervised ways: masked language model and next sentence prediction.

With the masked language mode, a sentence is taken from the training set. In the next step, about 15 percent of the words in that sentence are masked. For example, in the sentence

“I like to [mask1] a cup of coffee with [mask2] in the morning”

the words “drink” and “milk” have been masked. The model is then trained to predict the missing words in the sentence. The focus of the model is to understand the context of the words. The processing of the text data is no longer done in an unidirectional way from either left to right or right to left.

Using next sentence prediction as a training method, the model receives a pair of two sentences. The model's goal is to predict if the first sentence is followed by the second sentence. Therefore, the resulting model focuses mainly on how a pair of sentences are related.

Both models were trained together to minimize the combined loss function of the two strategies.

**Summary**

The use of NLP in computer science dates back to the 1950s. There is a wide range of application areas for NLP, which include topics such as question answering, sen- timent analysis, named entity recognition, and topic identification.

To be able to process language with computers, vectorization techniques such as Bag-of-Words, word vectors, and sentence vectors are used. However, these models come with some limitations. For example, if Bag-of-Words is used, we lose all infor- mation about word order. Therefore, this model can only be used if the word order

is not crucial. Moreover, some models, including BERT, have limitations towards the input length of a text (e.g., 256-word tokens). A larger paragraph of text can only be embedded using tricks, like segmenting it into smaller parts.

Nevertheless, there has been huge progress in NLP in the past years as computa- tional power has been increasing drastically and larger data corpora have become available to train language models.



# Unit 5

## Computer Vision

#### STUDY GOALS

On completion of this unit, you will be able to …

… define computer vision.

… explain how to represent images as pixels.

… distinguish between detection, description, and matching of features.

… correct distortion with calibration methods.

DL-E-DLBDSEAIS01-U05

1. Computer Vision

### Introduction

This unit will discuss the basic principles of computer vision. It starts with a definition of the topic, the historical background, and an overview of the most important com- puter vision tasks. After that, you will learn how an image can be represented as an array of pixels and how images can be modified using filters. We will illustrate how to detect features in images, such as edges, corners, and blobs. This knowledge will be used to illustrate how you can use calibration and deal with distortion.

Moreover, this unit addresses the topic of semantic segmentation, which can be used to classify pixels into categories.

### Introduction to Computer Vision

Computer vision is a topic that combines multiple disciplines. It is a mixture of com- puter science (especially artificial intelligence) and engineering (Wiley & Lucas, 2018). Computer vision tries to model human visual perception by processing and analyzing visual data. This data can, for instance, be static pictures or videos from cameras. The goal is to get a deep understanding of the visual aspects of the real world (Wiley & Lucas, 2018). Computer vision includes tasks, like the classification of objects or motion detection.

###### Historical Developments

Research in computer vision began in the 1960s at some of the pioneering universities for robotics and AI, such as Stanford University, the Massachusetts Institute of Technol- ogy, and Carnegie Mellon University. The goal of that early research was to mimic the visual system of humans (Szeliski, 2022). Researchers tried to make robots more intelli- gent by automating the process of image analysis using a camera attached to the com- puter. The big difference between digital image processing and computer vision at that time was that researchers tried to reconstruct the 3D structure from the real world to gain a better understanding of the scene (Szeliski, 2022).

Early foundations of algorithms, such as line labeling, edge extraction, object represen- tation and motion estimation date back to the 1970s (Szeliski, 2022). In the 1980s, there was a shift of focus towards the quantitative aspects of computer vision and mathe- matical analysis. Concepts, such as inference of shape from characteristics like texture, shading, contour models, and focus, evolved. In the 1990s, methods from **photogram- metry** were used to develop algorithms for sparse 3D reconstructions of scenes based on multiple images. The results led to a better understanding of camera calibration. Statistical methods, in particular eigenfaces, were used for facial recognition from pic- tures. Due to an increasing interaction between computer vision and computer graph-

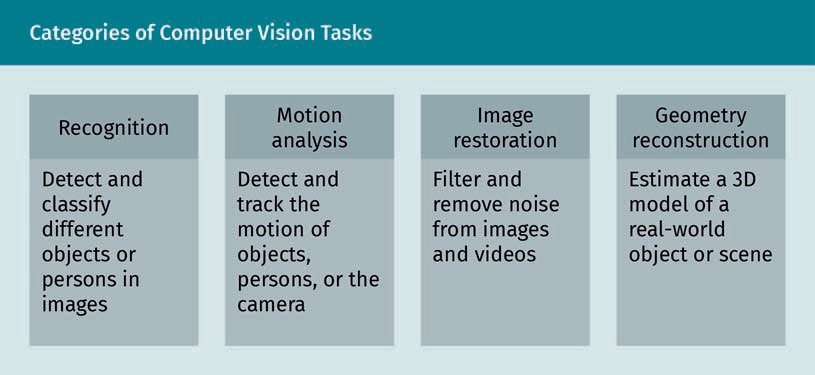
Computer Vision

ics, there has been a significant change in methods like morphing, image-based model- ing and rendering, image stitching, light-field-rendering, and interpolation of views (Szeliski, 2022).

Current developments go into the direction of optimization frameworks and machine learning approaches for those feature-based techniques. Further stimulation of the field of computer visions comes from the recent development in deep learning. These new methods outperform the classical methods on benchmark computer image data- sets in many tasks such as segmentation and classification or optical flow (O’Mahony *et al*., 2020).

###### Typical Tasks

There are four major categories in computer vision: recognition tasks, motion analysis, image restoration and geometry reconstruction. The following figure illustrates those tasks.



Recognition tasks

There are different types of recognition tasks in computer vision. Typical tasks involve the detection of objects, persons, poses, or images. Object recognition deals with the estimation of different classes of objects that are contained in an image (Zou *et al*., 2019). For instance, a very basic classifier could be used to detect whether there is a hazardous material label on an image or not. Making the classifier more specific could additionally recognize information about the label type such as “flammable” or “poi- son.” Object recognition is also important in the area of autonomous driving to detect other vehicles or pedestrians.

In object identification tasks, objects or persons that are in an image are identified using unique features (Barik & Mondal, 2010). For person identification, for example, a computer vision system can use characteristics, such as fingerprint, face or handwrit-

Photogrammetry

A group of contact- less methods to derive the position and shape of physi- cal objects directly from photographic images is called Photogrammetry.

ing. Facial recognition, for instance, uses biometric features from an image and com- pares them to the biometric features of other features from a given database. Person identification is commonly used to verify the identity of a person for access control.

Pose estimation tasks play an important role in autonomous driving. The goal is to esti- mate the orientation and/or position of a given object relative to the camera (Chen *et al*., 2020). This can, for instance, be the distance to another vehicle ahead or an obsta- cle on the road.

In optical character recognition (OCR), handwritten or printed text is recognized from an image and converted into a string, which can be processed by a machine (Islam *et al*., 2017). In online banking, for instance, OCR can be used to extract the relevant infor- mation for bank transfers such as the amount or the bank account information, from an invoice.

Motion analysis tasks

In classical odometry, motion sensors are used to estimate the change of the position of an object over time. Visual odometry, conversely, hand analyzes a sequence of images to gather information about the position and orientation of the camera (Aqel *et al*., 2016). Autonomous cleaning bots can, for instance, use this information to estimate the location in a specific room.

In tracking tasks, an object is located and followed in successive frames. A frame can be defined as a single image in a longer sequence of images, such as videos or anima- tions (Yilmaz *et al*., 2006). This can, for instance, be the tracking of people, vehicles, or animals.

Noise In computer vision, Noise refers to a quality loss of an image which is caused by a distur-

bed signal.

Image restoration tasks

Image restoration deals with the process of recovering a blurry or noisy image to an image of better and clearer quality. This can, for instance, be old photographs, but also movies that were damaged over time. To recover the image quality, filters like median or low-pass filters can remove the **noise** (Dhruv *et al*., 2017). Nowadays, methods from image restoration can also be used to restore missing or damaged parts of an artwork.

Geometry reconstruction tasks

In geometry reconstruction tasks, virtual 3D models of scenes from videos or images or even real world objects are estimated (Han *et al*., 2021). This is typically done based on multiple images that are taken from different perspectives.

###### Challenges in Computer Vision

In computer vision, there are five major challenges that must be tackled (Szeliski, 2022):

Computer Vision

* The illumination of an object is very important. If lighting conditions change, this can yield different results in the recognition process. For instance, red can easily be detected as orange if the environment is bright.
* Differentiating similar objects can also be difficult in recognition tasks. If a system is trained to recognize a ball it might also try to identify an egg as a ball.
* The size and aspect ratios of objects in images or videos pose another challenge in computer vision. In an image, objects that are further away will appear to be smaller than closer objects even if they are the same size.
* Algorithms must be able to deal with rotation of an object. If we look for instance at a pencil on a table, it can either look like a line when we look from the top or as a circle when we change to a different perspective.
* The location of objects can vary. In computer vision, this effect is called translation. Going back to our example of the pencil, it should not make a difference to the algorithm if the pencil is located on the center of a paper or next to it.

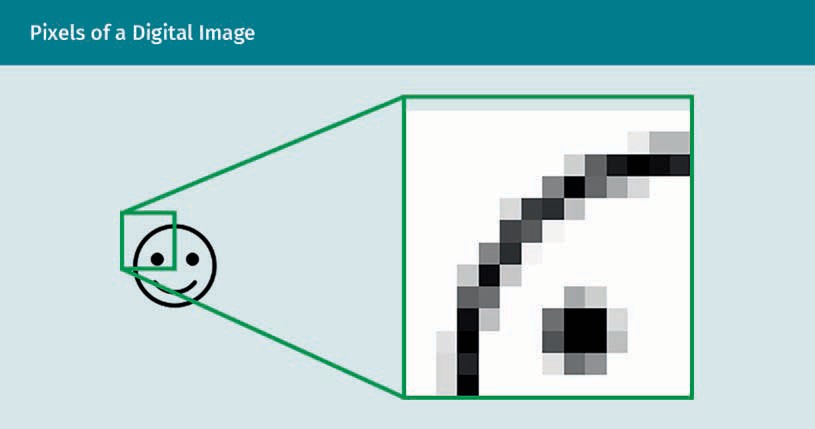
Because of these challenges, there is muchresearch towards algorithms that are scale-, rotation-, and/or translation invariant (Szeliski, 2022).

### Image Representation and Geometry

Computer vision is about processing digital images. To be able to process images with a computer, this section starts with an explanation of how to represent images in the form of numerical data. For this purpose, we introduce the concept of pixels. Subse- quently, we will address the topic of filters and how images can be modified using fil- ters.

###### Pixels

Images are constructed as a two-dimensional pixel array (Lyra *et al*., 2011). A pixel is the smallest unit of a picture. The word originates from the two terms “pictures” (pix) and “element” (el) (Lyon, 2006). A pixel is normally represented as a single square with one color. It becomes visible when zooming deep into a digital image. You can see an exam- ple of the pixels of an image in the figure below.



In the resolution of an image, the number of pixels is specified. If the resolution is high, the more details will be in the image. Conversely, if the resolution is low, the picture might look fuzzy or blurry.

Color representations

There are various ways to represent the color of a pixel as a numerical value. The easi- est way is to use monochrome pictures. In this case, the color of a pixel will be repre- sented by a single bit, being 0 or 1. In a true color image, a pixel will be represented by 24 bits.

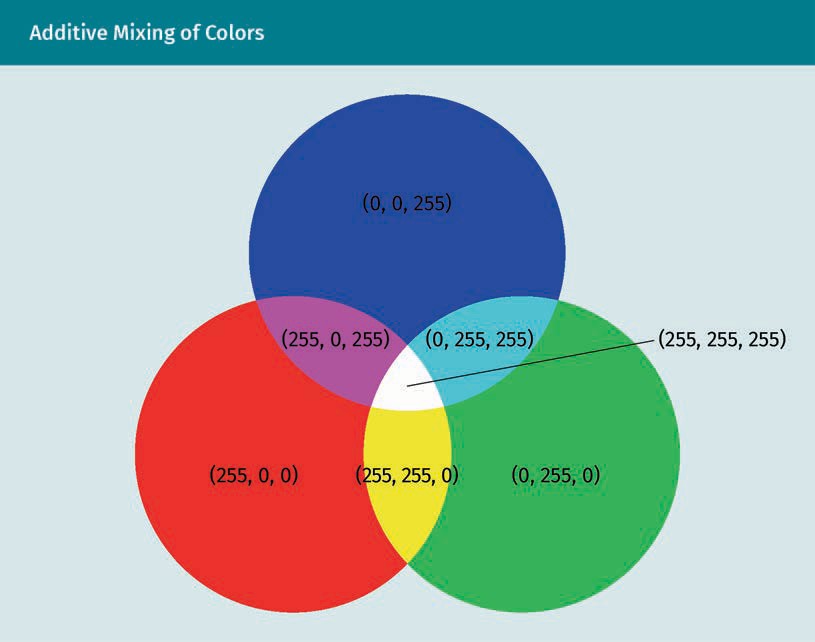
The following table shows the most important color representations with the according number of available colors (color depth).

|  |  |  |
| --- | --- | --- |
| **Color Representations in Images** | | |
| Name | Color representation | Color depth |
| Monochrome | 1 bit | 2 colors |
|  | 8 bit | 28 = 256 grayscale inten- sity levels or colors |
| Real color | 15 bit | 215 = 32.768 colors |
| High color | 16 bit | 216 = 65.536 colors |
| True color | 24 bit | 224 = 16.777.216colors |

Computer Vision

|  |  |  |
| --- | --- | --- |
| Name | Color representation | Color depth |
| Deep color | 30 – 48 bit | 230 − 248colors |

One way to represent colors is the RGB color representation. We illustrate this using the 24-bit color representation. Using RGB, the 24 bits of a pixel are separated in three parts, each 8 bits in length. Each of those parts represents the intensity of a color between 0 and 255. The first is the color red (R), the second green (G), and last blue (B). Out of these three components all the other colors can be mixed additively. For instance the color code RGB(0, 255, 0) will yield 100 percent green. If all values are set to 0, the resulting color will be black. If all values are set to 255 it will be white. The figure below illustrates how the colors are mixed in an additive way.



Another way to represent colors is the CMYK model. In contrast to the RGB representa- tion it is a subtractive color model comprised of cyan, magenta, yellow and key (black). The color values in CMYK range from 0 to 1. Therefore, to convert colors from RGB to CMYK, the RGB values first have to be divided by 255. Therefore, the values of cyan, magenta, yellow and key can be computed as follows:

K=1 − max R , G , B

255

1 − R − K

255

C= 1 − K

1 − G − K

255

M= 1 − K

1 − B − K

255

Y =

1 − K

255

255

While the RGB is better suited for digital representation of images, CMYK is commonly used for printed material.

Images as functions

We will now discuss how an image can be built from single pixels. To do that, we need a function that can map a two-dimensional coordinate (x,y) to a specific color value. On the x-axis we begin on the left with a value of 0 and continue to the right until the maximum width of an image is reached. On the y-axis, we begin with 0 at the top and reach the height of an image at the bottom.

Let us look at the function f x, y for an 8-bit grey scale image. The function values of f 42, 100 = 0 would mean that we will have a black pixel 42 pixels to the right and 100 pixels below the starting point. In a 24-bit image the result of the function would be a triple value indicating the RGB intensity of the specified pixel.

###### Filters

Filters play an important role in computer vision when it comes to applying affects to an image, implementing techniques like smoothing, or inpainting, or extracting useful information from an image, like the detection of corners or edges. It can be defined as a function that gets an image as an input, applies modifications to that image, and returns the filtered image as an output (Szeliski, 2022).

2D convolution

A frequently used technique to filter images is 2D convolution. If 2D convolution is applied to an image, a small matrix (also called a convolution matrix or kernel) is moved over the matrix of an image pixel by pixel and multiplied with the values of the matrix. The convolution matrix usually consists of 3x3 or 5x5 elements (Smith, 1997).

The convolution of an image I with a kernel k with a size of n and a center coordinate a

can be calculated as follows:

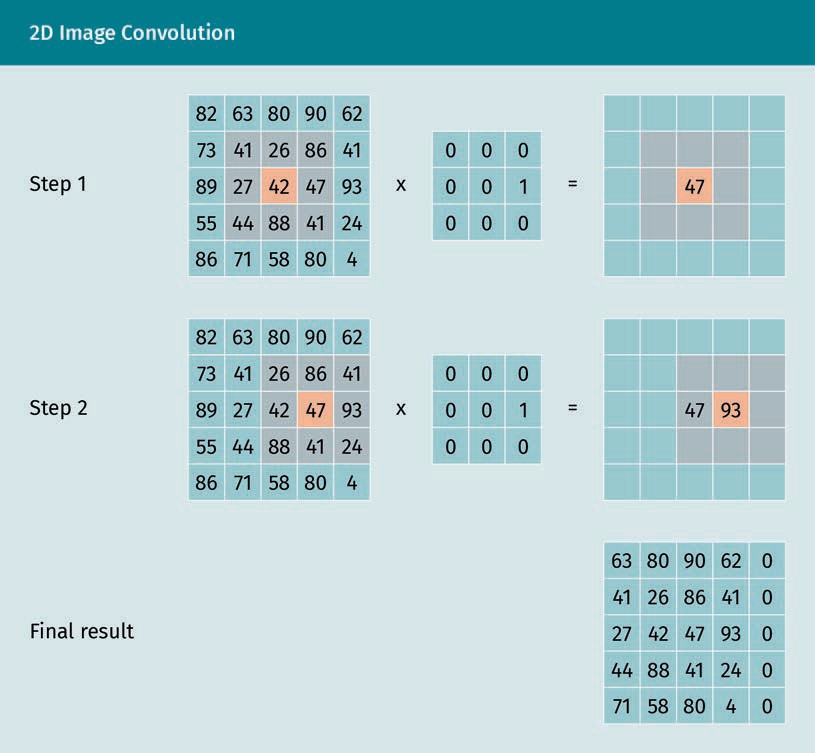
n n

I ⋅ x, y = ∑ ∑I x − 1 + a, y − j + a k i, j

i = 1 i = j

Computer Vision

where I · x, y is the value of the resulting image I · at position x, y while I is the original image. To understand the process, we will use the following example of a 3x3 convolution. The kernel matrix used for the convolution is shown in the middle column of the figure.



The kernel matrix is moved over each position of the input image. In our input image the current position is marked orange. In our example we start with the center position of the image and multiply the image on this position with the values of the kernel matrix. The resulting value for the center position of our filtered image is computed as follows:

0 · 41 + 0 · 26 + 0 · 86 + 0 · 27 + 0 · 42 + 1 · 47 + 0 · 44 + 0 · 88 + 0 · 41 = 47

In the next step, we shift the kernel matrix to the next position and compute the new value of the filtered image:

0 · 26 + 0 · 86 + 0 · 41 + 0 · 42 + 0 · 47 + 1 · 93 + 0 · 88 + 0 · 41 + 0 · 24 = 93

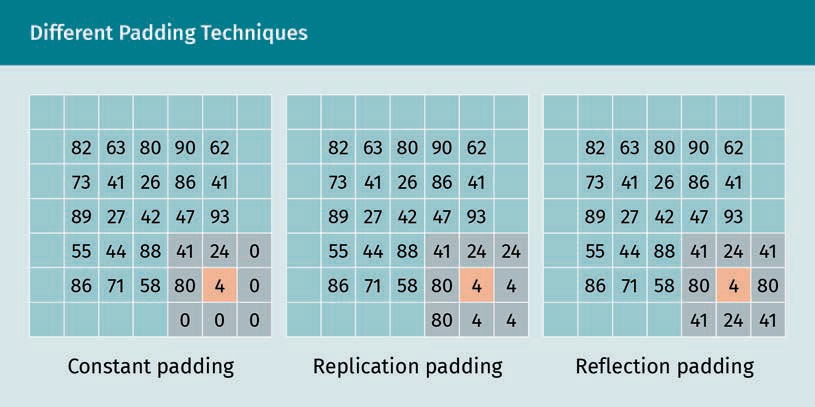
The bottom row in our figure shows the result after all positions of the image have been multiplied with the kernel matrix.

Padding techniques

If convolution techniques are applied to images, we face the problem that in the first and last rows and columns of an image there will not be enough values to apply the matrix multiplication with the convolution matrix. To solve this, we can add additional values at the border of our input images. This process is referred to as padding (Sze- liski, 2022).

There are three padding techniques that are commonly used: constant, replication, and reflection padding.

In constant padding, a constant number (e.g., zero) is used to fill the empty cells. Repli- cation padding uses a replication of the values from the nearest neighboring cells. In reflection padding, the value from the opposite side of a pixel is used to fill the cell. For instance, the cell on the top left will be filled with the value on the bottom right (Sze- liski, 2022).



The figure above illustrates how the three padding techniques are applied to an image.

###### Distortion

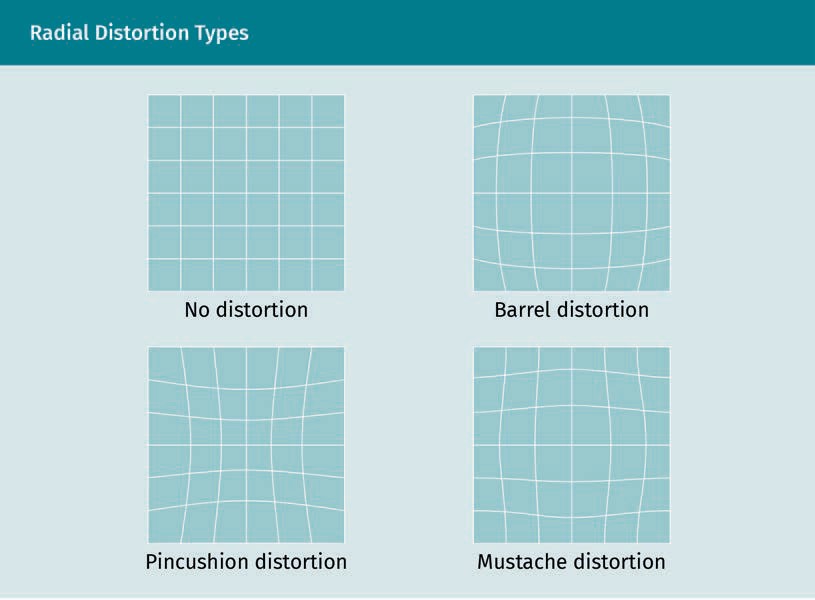
Image processing in computer vision is normally done with the assumption that an image we receive from a camera is a linear projection of a scene. That means that if we have a straight line in the real world we can expect it to be a straight line in the digital representation of the image (Szeliski, 2022). However, in practical scenarios camera len- ses often cause distortion. There exist two kinds of distortions – radial and tangential – that will be explained in the following.

Computer Vision

Radial distortion

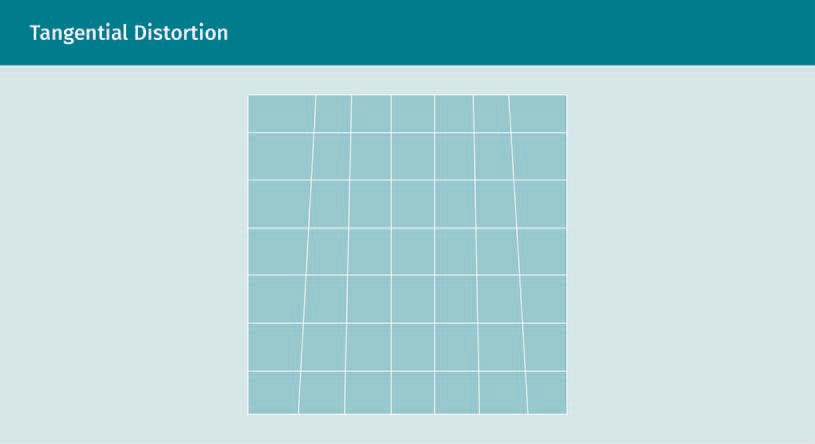
Radial distortion appears when lines that are normally straight bend towards the edge of the camera lens (Wang *et al*., 2009). The intensity of the distortion depends on the size of the lens. With smaller lenses we will find higher distortion. Moreover, radial dis- tortion is also more dominant when wide-angle lenses are used. In general, there are four types of radial distortion (Szeliski, 2022):

1. Barrel distortion/positive radial distortion: Lines in the center of an image are bent to the outside.
2. Pincushion distortion/negative radial distortion: Lines in the center of an image are bent to the inside.
3. Complex distortion/mustache radial distortion: Lines with a combination of positive and negative distortion.
4. Fisheye radial distortion: Occurs with ultra wide-angle lenses, e.g., a peephole.



Tangential distortion

Besides radial distortion, tangential distortion is another effect that can often be observed in digital imaging. Tangential distortion is caused if the image sensor unit and the camera lens are not properly aligned. If the camera lens and the image plane are not parallel, the distortions will look as shown in the graphic below.



To address distortion in digital image processing, mathematical models like the Brown- Conrady model (Brown, 1966) can be used to describe and correct the effects of the distortion. To be able to apply those models, it is important that the extrinsic and intrinsic parameters of the camera are known. These parameters can be determined by calibration.

###### Calibration

Camera calibration estimates the extrinsic and intrinsic parameters of a camera (Sze- liski, 2022). Only if these parameters are known can the camera be properly calibrated. The calibration makes it possible to extract distortion from the images.

Extrinsic characteristics of a camera are, for instance, the orientation in real world coordinates and the position of the camera. The intrinsic characteristics include parameters such as the optical center, the focal length, and the lens distortion parame- ters.

If the camera is calibrated properly, images can reliably be recovered from distortion which allows us, for instance, to measure distances and sizes on those images in units as meters and, therefore, reconstruct a 3D model of the underlying scenario from the real world.

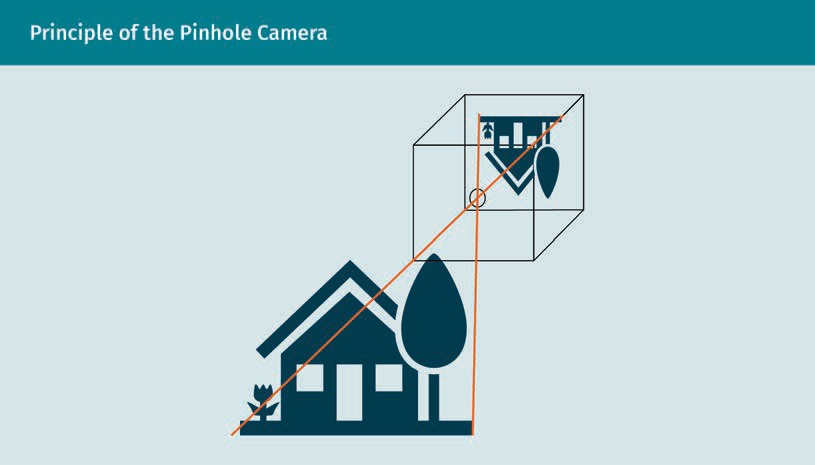
Pinhole camera The pinhole camera was the very first camera. It uses a box

Techniques

To be able to determine the calibration parameters of a camera, it is important to know the coordinates from the original real world 3D representation as well as the corre- sponding coordinates of the 2D image (Szeliski, 2022). A good example to illustrate the process of transferring a 3D image into a 3D image in a simplified model is the **pinhole camera**. In this process we use three coordinate systems:

Computer Vision

1. the 3D coordinate system of the camera
2. the 3D real world coordinate system
3. the 2D coordinate system of the projected image



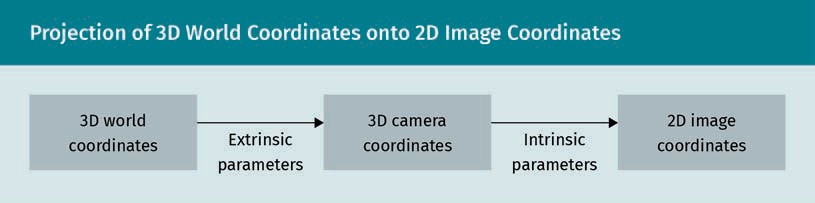
The projection process is done in two steps:

1. Transform the coordinates from the 3D world to the 3D camera coordinates. For this step, extrinsic parameters, such as rotation and translation of the information are used.
2. Transform the 3D camera coordinates to the 2D image coordinates. In this step, intrinsic parameters, such as focal length, distortion parameters, and optical center are applied.

To map the 3D coordinates from the real world to a two-dimensional image, a 3x4 pro- jection matrix (often referred to as a camera matrix) is used. When we multiply the 3D coordinates with this matrix, we will receive the 2D coordinates of the projected point on the image pane.

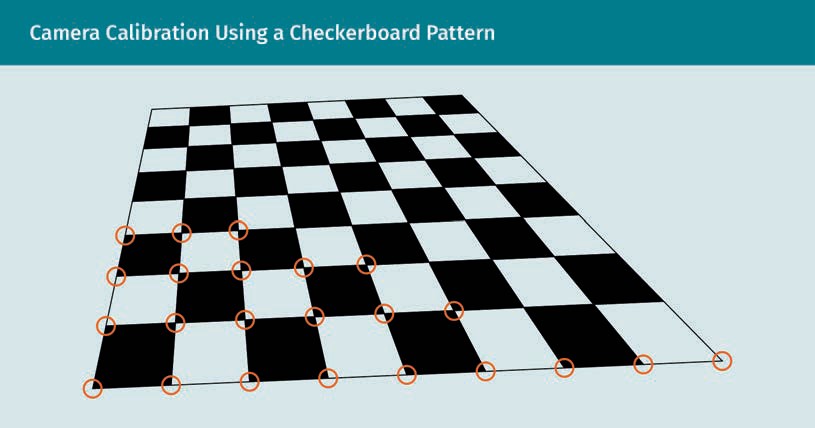
The figure below illustrates the steps of the projection process when 3D real world coordinates are transformed to the 2D image coordinates.

with a small pinhole to generate an image on the opposite side of the box.



To apply the projection steps illustrated above, we need to know the intrinsic and extrinsic camera parameters. These can be estimated using camera calibration. To understand the practical implementation of the calibration process we will look at flex- ible techniques for camera calibration (Zhang, 2000).

This technique uses two or more images as an input as well as the size of the object. A good object for camera calibration is, for instance, a checkerboard. After the calibration process, we will receive the extrinsic parameters rotation and translation and the intrinsic camera parameters optical center, focal length, and distortion.



The calibration process works as follows:

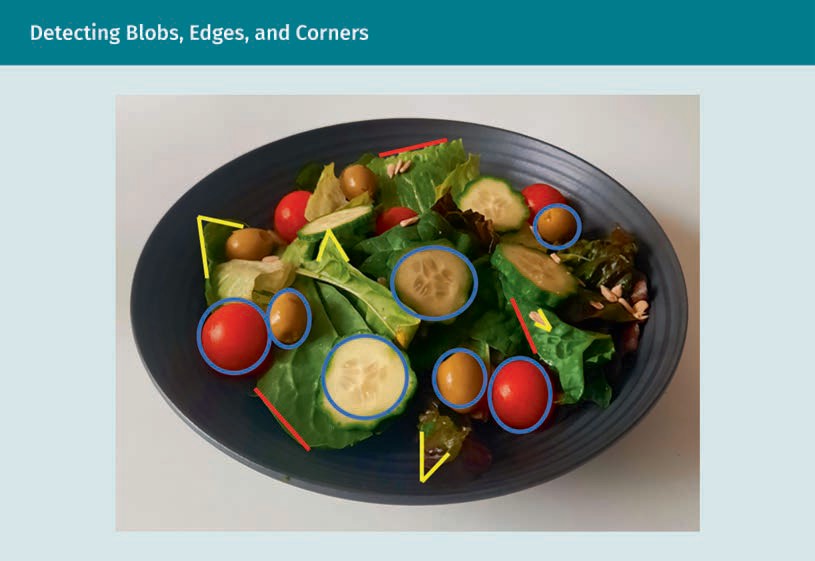
* 1. Select at least two sample images, which should be well-structured patterns, such as a checkerboard pattern.
  2. Identify distinctive points in each image. If we use a checkerboard pattern, this can, for instance, be the corners of the individual squares. Because of the clear structure of the checkerboard pattern with the black and white squares, the corners are easy to detect. They have a high gradient at the corners in both directions.
  3. Localization of the corners of the squares. For the checkerboard pattern this can be done in a very robust manner. To be able to identify the 3D coordinates of the cor- ners in the 3D real world, we need to know the size of the checkerboard and need two or more sample images. Moreover, we know the 2D coordinates of the corners in the image from the picture that was taken by the camera. Using this information, we can calculate the camera matrix and the distortion coefficients. The distortion coef- ficients can be used by applying the Brown-Conrady model (Brown, 1966).

Computer Vision

### Feature Detection

In the context of computer vision, features can be defined as points of interest of an image, which contain the required information to solve a respective problem (Hassa- ballah *et al*., 2016). To find those features in a picture, there exists a large variety of fea- ture detection algorithms. Once the features are detected, the semantic information about them can be extracted. The coordinates of a feature, i.e., on which position it is located in an image, is the feature keypoint. The semantic information extracted about a feature is stored in a vector, which is also called a feature descriptor or feature vector. The detection and extraction of features is often an important part of the preprocess- ing in machine learning applications. The extracted feature vectors can subsequently be used as an input for image classification. In motion tracking or recognition of indi- viduals or similar objects in multiple images, feature matching can be used.

The most common types of features are blobs, edges, and corners. Blobs are formed by a group of pixels that have some properties in common. Regions that differ in proper- ties belong to different blobs. This can, for instance, be different color or brightness compared to the areas surrounding a region. Edges are indicated by a significant change of the brightness of pixels. They can be identified by a discontinuity of the image intensity, i.e., a sudden change in the brightness of an image (Jain *et al*., 1995). Corners are the connection between two edges. The image below illustrates the difference between blobs (blue), edges (red), and corners (yellow).



If we want to detect all tomatoes in the picture, we can use an algorithm to detect all blobs. However, there will still be the challenge of distinguishing tomatoes from other round objects, like olives or cucumbers. This challenge can be tackled if we use a fea- ture description algorithm to extract the information that is characteristic of a tomato and construct a feature descriptor from this information. The feature descriptor could, for instance, include information about the surrounding n pixel values or the color of the pixels.

Once we have the feature descriptor for our cucumber candidate, it is possible to com- pare it with other feature descriptors from cucumber images using a feature matching algorithm. This feature matching algorithm allows us to detect all the cucumber slices in the image. As we have seen in our example, feature engineering is usually performed in three steps:

1. Feature detection
2. Feature description/extraction
3. Feature matching

Feature detection

To detect features such as edges or corners, there exist several methods. To detect edges in images, 2D convolution can be used. Edges are characterized by a significant difference of the pixel values to the surrounding pixels. If we look at an edge, there will be a clear difference in brightness and/or color compared to the surrounding pixels.

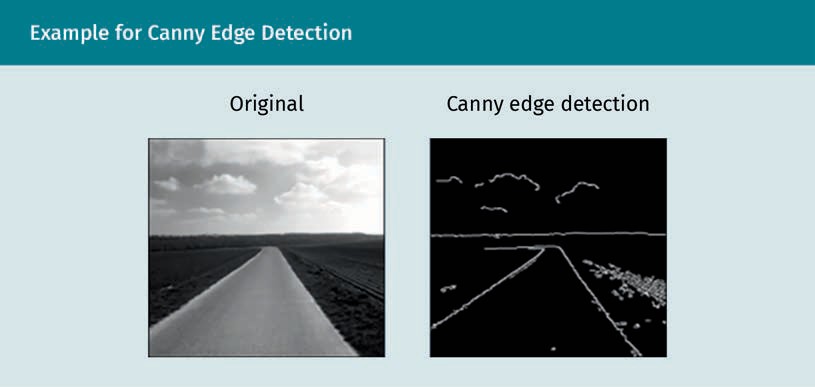


The figure above shows an example of edge detection. The edge between the road and the surrounding grass is clearly visibly in this example. On the upper left part of the zoomed in image we can see some variations of dark green colors, the lower right part is filled with variations of light grey. The edge separates both parts of the image.

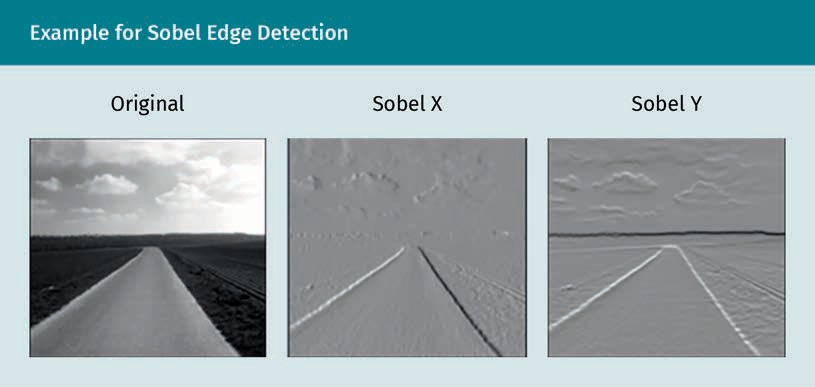
Two techniques that are commonly used for edge detection are the Canny edge detec- tor and the Sobel filter. The Canny edge detection (Canny, 1986) analyzes the change between pixel values. For this purpose, it uses the derivatives of the x and y coordi-

Computer Vision

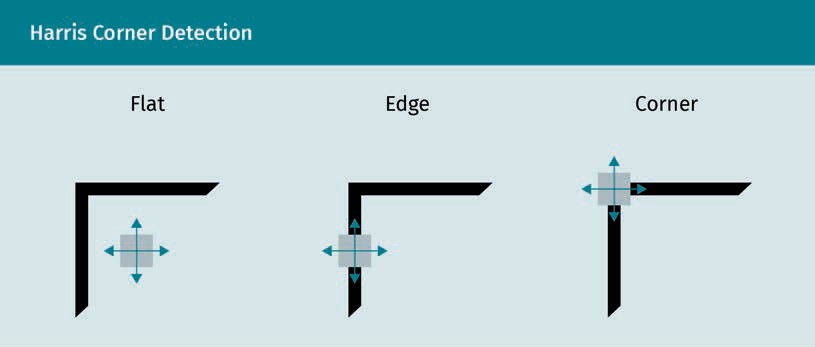
nates. The algorithm works with two–dimensional values, i.e., it works only on single color images such as grey scaled images. The figure below shows the result of the Canny edge detection in our example picture.



When using Sobel filters for edge detection, two special kernel matrices are used, one for each of the axes. These Sobel operators use convolution to transfer the original image into a gradient image. High frequencies in the gradient image indicate areas with the highest changes in pixel intensity which are likely to be edges. Therefore, in a sec- ond step, the algorithm is often combined with a threshold function to detect the images. The figure below shows the Sobel edge detection for the x and y direction.



For corner detection in images, one of the most prominent algorithms is the Harris cor- ner detection (Harris & Stephens, 1988). This algorithm analyzes the change of the pixel values in a sliding window that is moved in different directions. The sliding window can be as small as, for instance, 7x7 pixels. The figure illustrates how flat areas, edges, and corners can be detected using the sliding window technique.



The left image shows the window in a flat area with no edges or corners. In the under- lying window, there is no significant change in the values of the pixels if the window is moved into any direction. In the middle image, the window is moved on an edge but does not touch the other edge. This means we only have a change in a pixel value when we move the image in the horizontal direction. If we move the image in a vertical direction, there will be no changes in the pixel value. In the image on the right, the slid- ing window is moved over a corner. In this image, we will have a significant change in the pixel value no matter in which direction we move the image.

Therefore, if we want to detect corners, we have to find the window where the change of the underlying pixels is maximized in all directions. To formalize this idea mathemat- ically, Harris corner detection uses the Sobel operators which were explained previ- ously.

Feature description

For further processing of the features detected in the feature detection step, it is important to be able to describe those features in a way that a computer can use them and distinguish one from another. For this purpose, we use feature vectors/feature descriptors, which contain semantic information about the features. One possibility to describe features is the Binary Robust Elementary Features (BRIEF) algorithm (Calonder *et al*., 2010). To describe a feature, a binary vector is used.

The vector is constructed using an image patch which is constructed by comparing the intensity of a pair of pixels. In a first step, a patch p at position x is first smoothed. Afterwards the pixel intensity p x is computed. In a test τ the result of the comparison is coded into a binary value according to the following equation:

τ p; x, y : =

1 if p x < p y

0 otherwise

The major advantage of the BRIEF algorithm is that it is fast to compute and easy to implement. However, feature extraction for features that are rotated more than 35 degrees is no longer accurate (Hassaballah *et al*., 2016). Algorithms like Oriented FAST and Rotated BRIEF (ORB) try to overcome this limitation (Rublee *et al*., 2011).

Computer Vision

Another algorithm for feature description is the SIFT algorithm (Scale-Invariant Feature Transform) (Lowe, 1999). The SIFT algorithm has been enhanced by the SURF algorithm (Speeded-Up Robust Features) (Bay *et al*., 2008), which provides a performance improved variation of the SIFT algorithm. However, as both algorithms have been patented, they cannot be used as freely as for instance ORB. Additionally compared to ORB their accuracy is lower and the computational cost higher (Rublee *et al*., 2011).

Feature matching

The goal of feature matching is to identify similar features in different images. This could, for instance, be when detecting the same person in different scenarios. Feature matching is an important component in tasks like camera calibration, motion tracking, object recognition, and tracking.

One very simple technique for feature matching is brute force matching, which com- pares the feature descriptors of source and target image and computes the distance between those images (Jakubovic & Velagic, 2018). For numeric values of the feature vectors, we can use the Euclidean distance (Wang *et al*., 2005). For binary vectors, because they are generated when using the BRIEF algorithm, the Hamming distance is an appropriate approach to calculate the distance (Torralba *et al*., 2008).

Especially when dealing with large datasets and high dimensional feature vectors, Fast Library for Approximate Nearest Neighbors (FLANN) provides a more sophisticated method for feature matching. It contains a set of algorithms using a nearest neighbors search and has lower computational costs than brute force matching. The most appro- priate algorithm is automatically selected depending on the dataset. However, it is less accurate than brute force matching (Muja & Lowe, 2009).

###### Important Characteristics for Feature Detection and Extraction

According to Hassaballah *et al*. (2016), there are several characteristics a good algorithm for feature detection and extraction from images should have: robustness, repeatability, accuracy, generality, efficiency and quantity. The characteristics are explained in the table below.

|  |  |
| --- | --- |
| Important Characteristics of a Feature Detection and Extraction Algorithm | |
| Robustness | Reliable feature detection even under difficult conditions, such as different lighting conditions, noise (disturbed image signals), or changes in position, scale, or rotation of the feature |
| Repeatability | Replicability of feature detection independent from per- spective and angle |

|  |  |
| --- | --- |
| Important Characteristics of a Feature Detection and Extraction Algorithm | |
| Accuracy | Accurate localization of a feature in an image based on its pixel position |
| Generality | Application of feature detection and extraction in a differ- ent use case without additional adaption |
| Efficiency | Low computational costs |
| Quantity | Ability of the algorithm to detect (almost) all features present in an image to be able to generate a meaningful representation of that image |

###### Challenges in Feature Detection

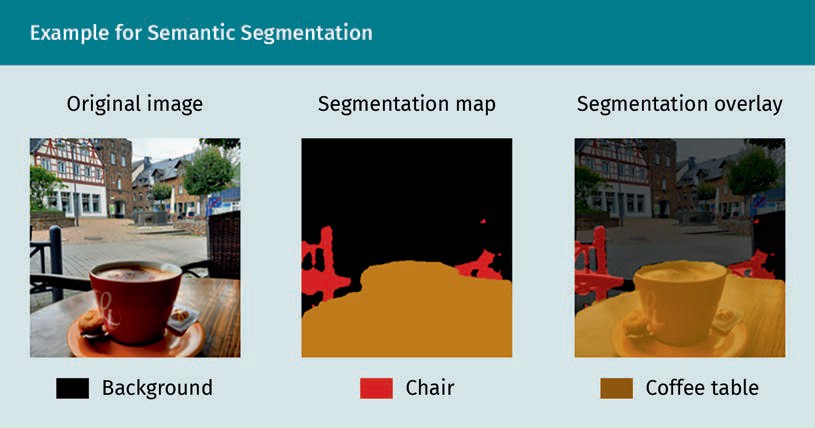
When performing feature detection and extraction on an image, there are several chal- lenges. While humans can easily identify objects no matter how they are located or lit, those differences can pose a great challenge for a computer. Therefore, there is still much ongoing research to develop algorithms that are less prone to factors, such as noise, varying lighting conditions, changes of camera perspectives, rotation or transla- tion of objects, and changes of scale.

### Semantic segmentation

In semantic segmentation, also known as image segmentation, parts of an image that belong to the same object class are put into the same cluster (Li *et al*., 2018; Thoma, 2016). The prediction is performed on a pixel-level, i.e., each pixel of an image will be classified according to its category.

To perform the semantic segmentation, the algorithm receives an image with one or more objects as an input, and outputs an image where each pixel is labeled according to its category. The figure below illustrates how semantic segmentation can be applied to an image. In the image, every pixel is either categorized as background, chair, or cof- fee table.

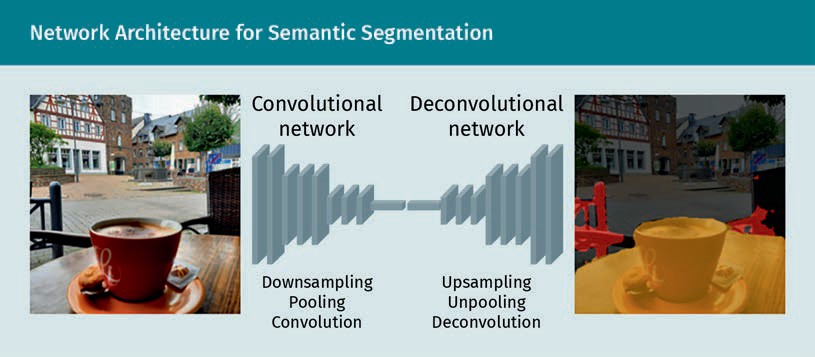
Computer Vision



###### Semantic Segmentation Techniques

Algorithms for semantic segmentation can be seen as a structured labeling problem on a pixel by pixel level and are often based on convolutional neural networks (CNNs) (Chen *et al*., 2016; Long *et al*., 2015).

If a CNN-based strategy is used, a network on top of the convolutional layers can be learned. This network consists of deconvolution and convolution layers. After training the network it can be applied to the proposals of the individual objects. The final semantic segmentation map is constructed as a combination of the results from the instance-wise segmentations (Noh *et al*., 2015). The architecture of the entire neural network is illustrated in the figure below.



The convolutional part of the network is used for feature extraction. It transforms the image from the input into a multidimensional representation of its features. The deconvolution network uses the features that have been extracted from the convolu-

Conditional random

fields An undirected prob- abilistic model that also considers neighboring samples for classification is known as a condi- tional random field.

tion network to generate the shapes of the object segmentation. Its unpooling and deconvolution layers are used to identify class labels based on the pixels and predict the segmentation masks. It generates a probability map as an output, which has the same size as the input image. For each pixel this probability map indicates the proba- bility of it belonging to one of the given classes (Noh *et al*., 2015). Additionally, to refine the label map, it is possible to apply fully connected **conditional random fields** to the output of the network (Krähenbühl & Koltun, 2012).

###### Use Cases

Semantic image segmentation can be helpful many use cases:

* Autonomous driving: detecting other vehicles, road lanes, pedestrians, or sidewalks (Kaymak & Uçar, 2019).
* GeoSensing: analyze information about land usage such as agricultural areas, for- ests, or areas of water from satellite images (Pollatos *et al*., 2020).
* Pose estimation/motion capture: identifying and tracking of body parts like legs, arms, head or eyes (Liu *et al*., 2013).
* Medicine: detecting of affected brain areas by tumors (Işın *et al*., 2016).

**Summary**

Computer vision is an interdisciplinary field that combines methods from computer science, engineering, and artificial intelligence. It dates back to the 1960s when researchers first tried to mimic the visual system of humans. Typical tasks in com- puter vision deal with topics such as recognition tasks image restoration, motion analysis, and geometry reconstruction.

In computer vision, images are represented using pixels. Models like the Brown- Conrady model can be used to address the distortion of digital images. Besides that, it is also important to know the calibration parameters of a camera to address radial and tangential distortion.

Feature detection algorithms in computer vision can be used to detect the points of interest. After the feature detection, the features are transformed into feature vec- tors, which can then be used for feature matching.

Using methods from semantic segmentation, the pixels of an image can be put into different categories to classify the content of an image.