What Makes Computing Intelligent?
A Tutorial and Practical Guide

Michael T. M. Emmerich
Faculty of Information Technology, University of Jyväskylä

26 August 2025

Abstract

Computing has evolved from fixed, automated pipelines to systems that learn, reason, and adapt. This tutorial introduces the fundamental ideas that make computing *intelligent*, organizes the field into practical categories, and presents two complementary organizing frames: (i) four practical cornerstones that enable AI in practice (data, compute, algorithms, interfaces) and (ii) five methodological pillars (reasoning, learning, representation, self-adaptation, interaction). The tutorial is designed for students and practitioners and includes short exercises and a capstone ideation activity around a "Berry Assistant" application.

Contents

1 Introduction

What is Artificial Intelligence (AI)? In this tutorial, we regard AI as the study and engineering of computer systems that perform complex tasks typically requiring human intelligence (cf. [**16**]). Beyond being fast or automated, *intelligent* computing systems can exploit structure, learn from data and feedback, reason with domain models, represent knowledge at appropriate levels of abstraction, adapt to new tasks, and interact effectively with humans and their environments.

Motivation and audience. This tutorial is intended for curious lay readers and young researchers who want a compact *map* of modern AI: what problems it addresses, which core ideas it builds on, and which *resources* (data, compute, algorithms, and interfaces) are typically required to make systems work reliably at scale. Our goal is to demystify the moving parts, show

how the methodological pillars connect to practical engineering, and provide enough pointers to the literature to support deeper study ([**16,17,18,21**]).

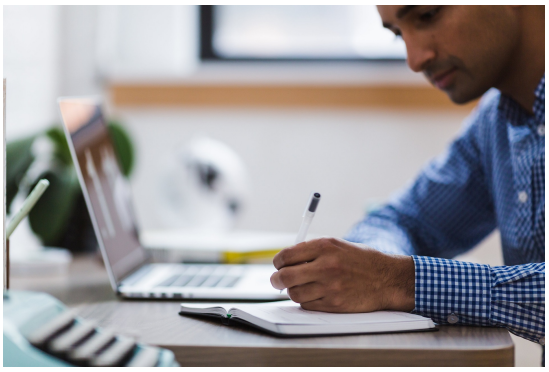

Figure 1: AI at the intersection of computation, data, and human problem solving. *Image credit:* `https://pixabay.com/fi/photos/mies-kirjoittaminen-2562325`.

2 Definitions and Categories of AI

Working definition. *Artificial Intelligence (AI)* denotes computer systems performing complex tasks typically requiring human intelligence ([**16**]).

Example categories. AI spans strategic gameplay such as chess, Go, and other combinatorial puzzles; predictive and prescriptive analytics in domains like finance and medicine; design and planning problems including routing, scheduling, drug discovery, and engineering design; language processing tasks such as spam filtering, translation, coding support, and conversational agents; and perception-and-control scenarios for autonomous robots, drones, and computer vision systems. (Overview texts: [**16,17,21**]; statistical learning foundations: [**20,32**].)

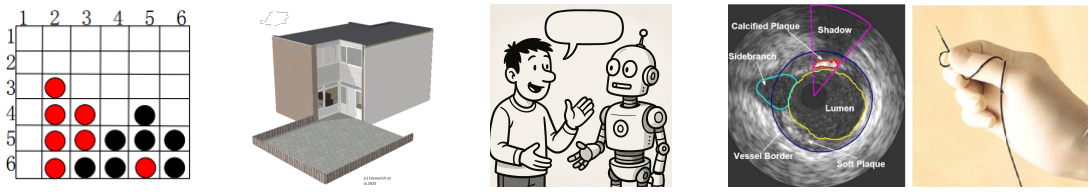

Figure 2: Examples across AI application areas. Strategic game play, computer-aided design, natural language processing, intelligent devices (drones, medical imaging).

Selected references: For concrete illustrations, see [**3**] on AlphaZero-like reinforcement learning for Morpion Solitaire, [**4**] on self-adaptive vision agents using evolutionary strategies, and [**5**] on drone monitoring and machine learning.

Exercises (Foundations and Categories).

- **E1.** Provide **three examples of AI systems** you use or know well. *Categorize* each (e.g., perception, decision support, control, dialogue) and explain *why* it qualifies as AI.
- **E2.** Compare your categorizations with a peer's and discuss which examples are *narrow* vs. potentially *general* and what would be required to broaden their scope.

3 Four Practical Cornerstones Enabling Today's AI

We emphasize four practical cornerstones that jointly enable modern AI: **Data**, **Compute Power**, **Algorithm Design**, and **Interfaces** (human and environmental). Each can be weak or strong in a given system, and characteristic profiles distinguish families of AI solutions (e.g., self-play systems for Go vs. conversational agents).

Data. High-quality data determines both the ceiling and the speed of learning. Useful datasets are *representative*, *curated* to remove noise and bias, *well-labeled* when supervision is required, and often *augmented* to improve generalization. For unsupervised or self-supervised pipelines, coverage and diversity matter more than labels; for safety-critical applications, provenance and documentation are essential ([**17,18,25**]).

Compute power. Modern AI relies on parallel compute (GPUs/TPUs) and memory bandwidth to train over large datasets and model families. Through batching and distributed training, compute determines the feasible model size and the timescale of iteration. Practical systems balance *cloud* resources for training with *on-device* or edge constraints for inference (latency, energy).

Algorithm design. Algorithms turn data and compute into capability: optimization methods (from gradient descent to evolutionary search), model classes (probabilistic models, kernel machines, neural networks), search and planning, and reinforcement learning. Design choices encode prior knowledge, control capacity, and affect sample/compute efficiency ([**16,17,18,20,21,34,35**]).

Interfaces. Interfaces connect AI to the world: data pipelines and sensors, user interfaces and APIs, and *human-in-the-loop* mechanisms for feedback, oversight, and preference learning. Clear interfaces also support *responsible* deployment (transparency, alignment) and efficient data acquisition (e.g., learning loops) ([**26,14,36**]).

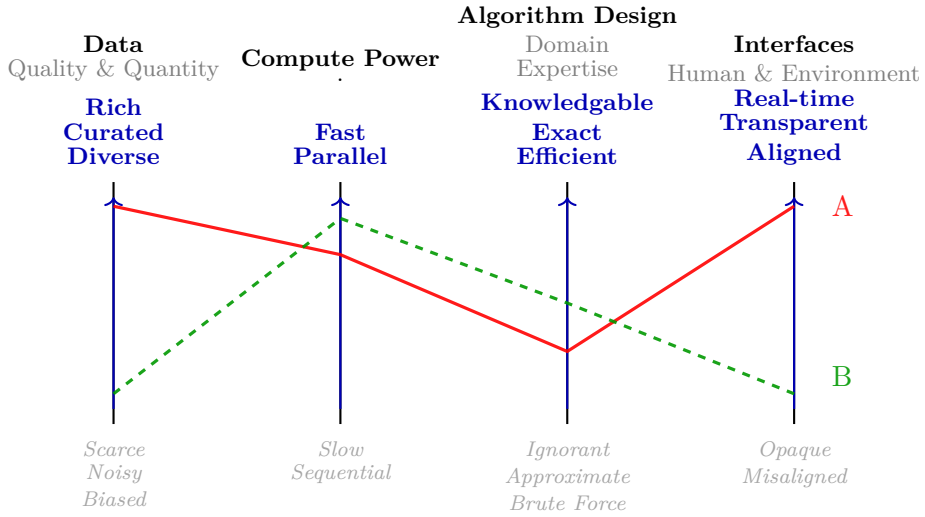

Figure 3: Four cornerstones of practical AI with two contrasting profiles (A, B), e.g., AlphaZero-style systems vs. chatbot-style systems.

References: [**1**]; [**2**].

Exercises (Cornerstones in Practice).

C1. For one of your examples from E1, sketch a **cornerstone profile**: rate Data, Compute, Algorithms, and Interfaces on a 1–5 scale and justify your ratings.

C2. Propose **two trade-offs** (e.g., higher accuracy vs. latency; transparency vs. performance) and suggest design choices to navigate them.

C3. Identify a **bottleneck** (data, compute, algorithms, or interfaces) and outline a plan to mitigate it (e.g., active learning, quantization, human-in-the-loop).

4 Five Core Methodological Pillars

We structure the tutorial around five methodological pillars: **Reasoning**, **Learning**, **Representation**, **Self-Adaptation**, and **Interaction**.

4.1 Representation

What it is. Encoding data and knowledge (symbols, rules, ontologies, feature vectors, neural activations). Trends move from manual features to end-to-end deep representation learning, connecting to self-adaptation (deep learning overviews: **[18,25]**; neural network/textbook treatments: **[17,31]**; probabilistic representations: **[22,37]**).

Why it matters. Proper abstraction reduces complexity and enables effective reasoning and learning.

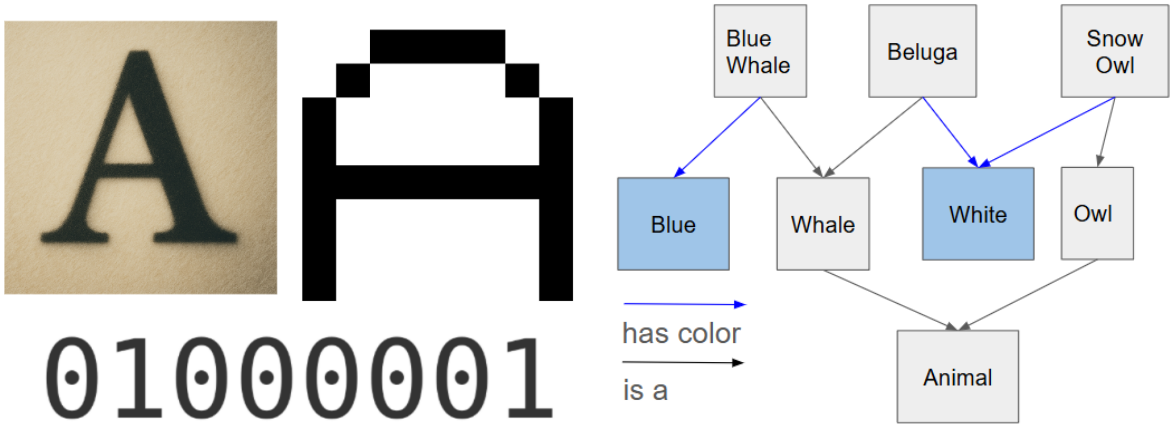

Figure 4: Examples of representational choices: learned features and explicit ontologies.

Reference: **[6]**.

Exercises (Representation for Plant Identification).

R1. List **features** useful for plant ID in an app (e.g., leaf shape, margin, venation, flower color, fruit type, habitat, seasonality, toxicity). Explain why each feature helps disambiguation.

R2. Propose a **knowledge graph schema**: define node types (species, genus, habitat, trait, image) and edge types (`has_trait`, `grows_in`, `similar_to`). Sketch a small example with 5–10 nodes.

R3. Describe a **hybrid pipeline** that combines image embeddings from a CNN with symbolic constraints from your knowledge graph to improve robustness.

4.2 Reasoning

What it is. Drawing conclusions by combining *rules*, domain models, and *facts* using symbolic and numerical inference (**[22,36]**). For example, from $A \to B$, $B \to C$ and the meta-rule $(x \to y \wedge y \to z) \Rightarrow x \to z$ we infer $A \to C$. Another example is to exploit mathematical models in

physics for numerical inference, such as Newton's law of motion $\vec{F} = m\vec{a}$ when reasoning about a control strategy for robotics. Challenges include combinatorial explosion (e.g., heuristic search [**38**]), uncertainty propagation, and imprecision (fuzzy logic).

Why it matters. To maximize the use of both data and expert knowledge/models.

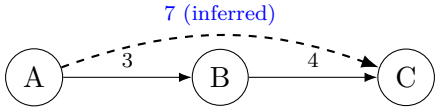

Figure 5: Reasoning composition: chaining implications.

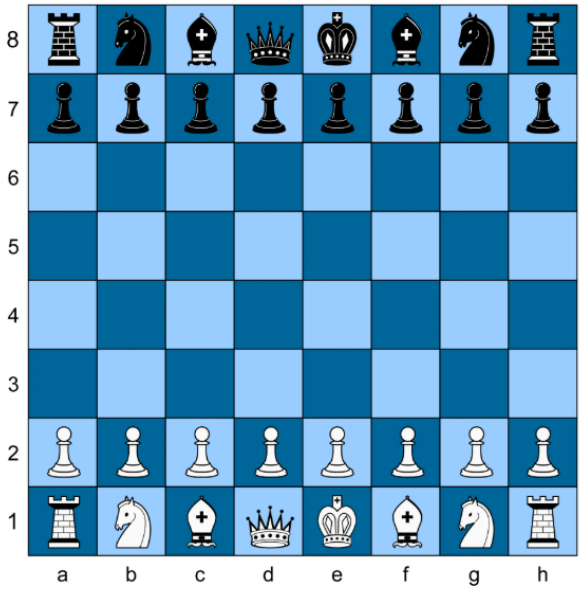

Figure 6: Reasoning and search: from classical planning to game-playing systems such as *Deep Blue*.

Reference: [**7**].

Exercises (Reasoning).

G1. Construct a **rule chain** of at least three implications in a real-world setting (e.g., *if it rains* $\rightarrow$ *road is wet*, *if road is wet* $\rightarrow$ *traffic slows down*, therefore *if it rains* $\rightarrow$ *traffic slows down*). Discuss the assumptions behind your chain.

G2. Introduce **uncertainty**: assign probabilities or fuzzy truth values to your rules and show how they affect the final inference.

G3. Compare symbolic inference (rules, logic) with numerical inference (statistics, probabilities) in your example. What are the strengths and limitations of each?

4.3 Learning

What it is. Supervised learning reduces error on selected training examples via (multiobjective) optimization; unsupervised learning explores data to reveal hidden patterns; learning requests informative data; transfer learning reuses knowledge across tasks; imitation learning learns from demonstrations; and reinforcement learning learns by trial-and-error with rewards.

Supervised learning. Seek a mapping $x \mapsto y$ from labeled pairs (x, y) by minimizing a task-specific loss for classification or regression. In practice one balances fit and generalization

Figure 7: Learning paradigms: playful (unsupervised) and supervised training. (Right image credit: `https://www.vecteezy.com/photo/68678450`)

(regularization, early stopping), chooses model classes (e.g. linear models, kernels, deep nets), and may optimize multiple objectives (accuracy, sparsity, latency) ([**17,21,20,32**]).

Unsupervised learning. Discover structure in unlabeled data via clustering, density estimation, and dimensionality reduction. Typical objectives include compactness and separation of clusters, likelihood maximization, and information preservation; examples include k-means, Gaussian mixtures, PCA, autoencoders, and VAEs ([**17,23,24**]).

Active learning. When labels are expensive, the learner selects the most informative data points to query (uncertainty sampling, expected model change), closing the loop between model and annotator to reduce labeling cost while maintaining quality ([**26**]).

Transfer learning. Reuse knowledge from a source task/domain to accelerate or improve a target task. Mechanisms include representation transfer (frozen backbones with small heads), parameter transfer (fine-tuning), domain adaptation, and multi-task learning ([**27,5**]).

Imitation learning. Learn a policy from demonstrations: either by behavior cloning (supervised learning on state–action pairs) or by inferring the demonstrator's objective and optimizing it (inverse reinforcement learning). Imitation can seed policies that are later refined with environment interaction ([**29,30,28**]).

Reinforcement learning. Learn to act by trial-and-error, optimizing expected cumulative reward through interaction with an environment. Core ideas include value functions, policy gradients, exploration, and credit assignment; self-play is a powerful driver for domains like games ([**19,1**]).

Developmental remark. A two-year-old can already pick berries: her competence probably emerges as a result of different learning styles—*unsupervised play* with small objects (sensorimotor exploration), learning by reward (reinforcement learning), *learning by imitation* (parents and older siblings as demonstrators), and occasional *active queries* (pointing/asking) that elicit targeted feedback, e.g., asking if a berry is good.
References: [**8**], [**9**].

Exercises (Supervised Learning: Finnish Vocabulary). Task. Treat vocabulary acquisition as a supervised learning problem. Given inputs (English words) and gold labels (Finnish translations), your goal is to *minimize the error* between your predictions and the labels.

Figure 8: Child searching and picking blueberries.

Instructions.

1. Write your predicted Finnish translation for each English word in the middle column.
2. After checking the gold labels, compute the 0–1 loss for each item (1 if incorrect, 0 if correct) and sum to obtain the total error.
3. Repeat after study; your aim is to reduce the total error over iterations ("epochs").

English (input)	**Your prediction (Finnish)**	**Gold label**
Teacher	_____	opettaja
Example	_____	esimerkki
Supervisor	_____	ohjaaja
Learning	_____	oppiminen
Computer	_____	tietokone
Knowledge	_____	tieto
Neural Network	_____	neuroverkko
Deep Learning	_____	syväoppiminen
Artificial Intelligence	_____	tekoäly
Intelligence	_____	älykkyys

Extension. Design an *active learning* loop: mark which words you are least confident about and query a native speaker or dictionary first. Track accuracy over time.

Scoring

For $n = 10$ items with predictions $\hat{y}_i$ and labels y_i, compute $L = \sum_{i=1}^{n} \mathbf{1}[\hat{y}_i \neq y_i]$. Your objective is to minimize L. Optionally report accuracy $(1 - L/n)$.

Optional reflection

What features or mnemonics helped you generalize? Could transfer learning help (e.g., reusing knowledge from other languages)?

4.4 Self-Adaptation and Meta-Learning

What it is. Handling new tasks via meta-learning and transfer; self-play and reward-driven learning; simulated evolution via selection–mutation–crossover; and transfer learning (learning task A helps to learn task B) ([**19,33,35,34**]).

Why it matters. Flexibility for changing environments and open-ended evolution.

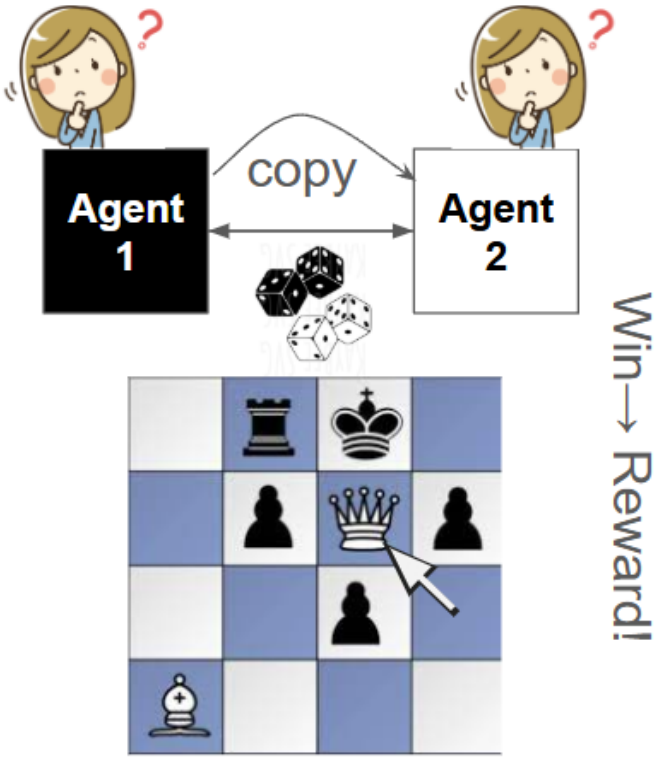

Figure 9: Self-play & deep reinforcement learning as drivers of emergent competence.

References: **[10]**; **[11]**; **[12]**; **[13]**.

Exercises (Self-Adaptation and Open-Ended Evolution).

- **S1.** Consider a **berry-picking robot swarm** capable of self-replication, with survival proportional to picking efficiency. List **three emergent risks** (e.g., ecological damage, resource overuse, adversarial behaviors) and why they might arise.
- **S2.** Propose **safeguards** (simulation-only evolution, containment, fitness shaping, human oversight, kill-switches) and discuss trade-offs with performance and exploration.
- **S3.** Write a short **ethical reflection**: which activities should be restricted to simulation vs. allowed in the real world, and under what regulatory framework?

4.5 Interaction (with Humans, Environment, Other Agents)

What it is. Mechanisms that connect perception, decision making, and action in the world, and that coordinate with *people* who may hold different goals, preferences, and perspectives ([**14,16**]).

Why it matters. Interaction turns learned capability into real-world usefulness and safety: systems must perceive in real time, act reliably, and *work with* stakeholders (users, operators, bystanders), supporting oversight and value alignment ([**14**]).

Low-level perception and control. At the sensorimotor layer, systems perform *real-time signal processing* (filtering, detection), *segmentation* and *interpretation* (e.g., object/scene understanding), and *state estimation* via probabilistic inference; these feed closed-loop controllers that map signals to actions (from reactive policies to planning-based control). Typical toolkits include deep perception models ([**25,18**]), probabilistic reasoning for fusion and tracking ([**22**]),

and search/planning to connect goals with feasible trajectories ([**38**]). Reinforcement learning policies may replace or complement classical controllers when rich interaction data are available ([**19**]).

High-level human-centered interaction. Above the control loop, systems coordinate with humans who differ in *goals*, *preferences*, and *perspectives* on a problem. This includes dialogue and explanation, *preference elicitation* and interactive learning (human-in-the-loop/active learning), and mechanisms for resolving trade-offs among competing objectives in context ([**14,26,16**]). Good interfaces capture feedback efficiently, reveal uncertainty, and enable meaningful oversight—linking back to the *Interfaces* cornerstone.

Figure 10: Interaction-centric systems: sensing, acting, and aligning with human goals. (Yelizaveta Thomashevska, licensed from iStock.com)

References: [**14**]; [**15**].

Exercises (Interaction and Safe Berry Picking).

- **I1.** Design a robot to **pick a berry without damaging the plant**. List the **interaction hardware** you would use (e.g., soft gripper or suction end-effector, tactile/force sensors, high-resolution RGB-D camera, hyperspectral snap for ripeness, compliant actuators) and justify each choice.
- **I2.** Sketch a **perception–action loop**: perception modules, state estimation, grasp planning, force control, safety checks, and human override.
- **I3.** Propose **evaluation metrics** (bruise rate, detach force peak, success rate, cycle time, plant damage score) and an experiment design to compare two grippers.

5 Workshop: Designing a "Berry Assistant"

Five-Minute Ideation (Flinga Whiteboard)

Imagine an AI app for **berry search, identification, and picking**, and collaboratively capture ideas (e.g., via Flinga) by specifying the resources needed in terms of the four cornerstones (data, compute, algorithms, and interfaces), by explaining how the five pillars (reasoning, learning, representation, self-adaptation, and interaction) are embodied in the design, and by outlining how the app will be **useful**, **reliable**, and **ethical**.

Figure 11: Collaborative ideation (left: access QR/link); concept mockup (right). **Short link:** `https://tinyurl.com/2tu2x7kx`.

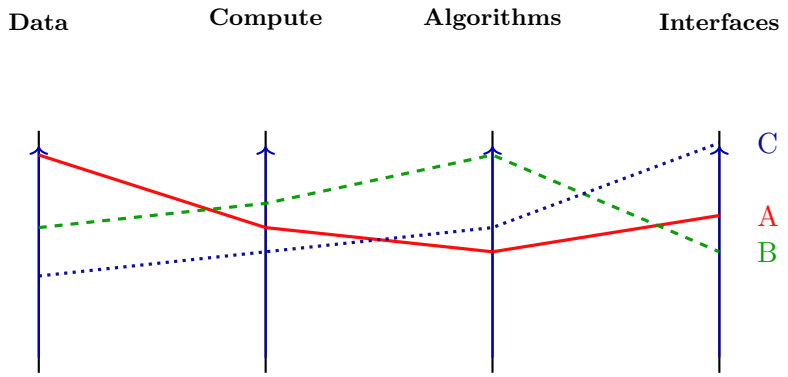

Figure 12: Suggestion for a cornerstone profiles for different Berry Assistant configurations (A) Finding on Map, (B) Identifying, and Picking Robot (C).

Cornerstones for the Berry Assistant

6 Conclusion

We presented a compact tutorial unifying practical enablers (cornerstones) with methodological pillars. The *Berry Assistant* ideation exercise illustrates translating the framework into a real application. For coursework or self-study, repeat the exercise for an AI application of your choice and prepare a short presentation.

Acknowledgments

Thanks to collaborators and students whose projects underpin several examples.

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, *et al.* (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676), 354–359.

[2] H. Wang, M. T. M. Emmerich, M. Preuss, & A. Plaat (2019). Alternative loss functions in AlphaZero-like self-play. In *2019 IEEE SSCI* (pp. 155–162). IEEE.

[3] H. Wang, M. Preuss, M. T. M. Emmerich, & A. Plaat (2020). Tackling Morpion Solitaire with AlphaZero-like ranked reward reinforcement learning. *SYNASC*. IEEE.

[4] T. Bäck, R. Li, J. Eggermont, M. T. M. Emmerich, E. Bovenkamp, J. Dijkstra, & J. Reiber (2016). Self-Adaptive Vision Agents Using Evolutionary Strategies. Human Competitive Design Competition, ACM-GECCO (3rd Place).

[**5**] A. Dmytriv, M. T. M. Emmerich, *et al.* (2020). Drone Monitoring and Machine Learning Technology. *MoMLeT+DS*, Lviv, Ukraine.

[**6**] H. Wang, I. Schwab, & M. T. M. Emmerich (2015). Comparing knowledge representation forms in empirical model building. *INTELLI 2015*, 184.

[**7**] M. T. M. Emmerich (2000). An interval constraint propagation technique for chemical process network synthesis. In *Proc. Knowledge Based Computer Systems (KBCS 2000)* (pp. 470–481). Mumbai.

[**8**] J. Zhao, L. Jiao, S. Xia, V. B. Fernandes, I. Yevseyeva, Y. Zhou, & M. T. M. Emmerich (2018). Multiobjective sparse ensemble learning by means of evolutionary algorithms. *Decision Support Systems*, 111, 86–100.

[**9**] A. Gopnik, A. N. Meltzoff, & P. K. Kuhl (1999). *The Scientist in the Crib: Minds, Brains, and How Children Learn*. William Morrow.

[**10**] H. Wang, M. T. M. Emmerich, M. Preuss, & A. Plaat (2023). Analysis of hyper-parameters for AlphaZero-like Deep Reinforcement Learning. *International Journal of Information Technology & Decision Making*, 22(02), 829–853.

[**11**] D. Silver, S. Singh, D. Precup, & R. S. Sutton (2021). Reward is Enough. *Artificial Intelligence*, 299, 103535.

[**12**] S. Meyer-Nieberg, & H. G. Beyer (2007). Self-Adaptation in Evolutionary Algorithms. In *Parameter Setting in Evolutionary Algorithms*. Springer.

[**13**] J. P. Bharadiya (2023). Transfer learning in natural language processing (NLP). *European Journal of Technology*, 7(2), 26–35. `https://doi.org/10.47672/ejt.1490`

[**14**] C. Bartneck, T. Belpaeme, F. Eyssel, T. Kanda, M. Keijsers, & S. Šabanović (2024). *Human-Robot Interaction* (2nd ed.). Cambridge University Press.

[**15**] K. Eyvindson, D. Burgas, C. Antón-Fernández, J. Hakanen, M. T. M. Emmerich, J. Klein, *et al.* (2024). MultiOptForest: An interactive multi-objective optimization tool for forest planning and scenario analysis. *Open Research Europe*, 3, 103.

[**16**] S. J. Russell & P. Norvig (2021). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

[**17**] C. M. Bishop (2006). *Pattern Recognition and Machine Learning*. Springer.

[**18**] I. Goodfellow, Y. Bengio, & A. Courville (2016). *Deep Learning*. MIT Press.

[**19**] R. S. Sutton & A. G. Barto (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press.

[**20**] V. N. Vapnik (1998). *Statistical Learning Theory*. Wiley.

[**21**] T. M. Mitchell (1997). *Machine Learning*. McGraw–Hill.

[**22**] J. Pearl (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.

[**23**] D. P. Kingma & M. Welling (2013). Auto-encoding variational Bayes. *arXiv:1312.6114*.

[**24**] G. E. Hinton & R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.

[**25**] Y. LeCun, Y. Bengio, & G. Hinton (2015). Deep learning. *Nature*, 521(7553), 436–444.

[**26**] B. Settles (2012). *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning, 6(1), 1–114. Morgan & Claypool.

[**27**] S. J. Pan & Q. Yang (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359.

[**28**] S. Ross, G. J. Gordon, & J. A. Bagnell (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)* (PMLR 15, pp. 627–635).

[**29**] S. Schaal (1997). Learning from demonstration. In *Advances in Neural Information Processing Systems 9 (NIPS 1996)* (pp. 1040–1046). MIT Press.

[**30**] P. Abbeel & A. Y. Ng (2004). Apprenticeship learning via inverse reinforcement learning. In *ICML*.

[**31**] C. M. Bishop (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[**32**] N. Cristianini & J. Shawe-Taylor (2000). *An Introduction to Support Vector Machines*. Cambridge University Press.

[**33**] J. H. Holland (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

[**34**] K. Deb, A. Pratap, S. Agarwal, & T. Meyarivan (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197.

[**35**] H.-G. Beyer & H.-P. Schwefel (2002). Evolution strategies – A comprehensive introduction. *Natural*

Computing, 1, 3–52.

[36] J. Pearl (2009). *Causality* (2nd ed.). Cambridge University Press.

[37] N. Friedman, D. Koller, & A. Pfeffer (1999). Learning probabilistic relational models. In *IJCAI*.

[38] P. Hart, N. Nilsson, & B. Raphael (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107.