



Part on Theoretical Fundamentals - subjects:

Information Theory (guarantor Ivo Bukovsky)

1. Probability in information theory (conditional probability, joint probability), accuracy and robustness of machine learning classifiers, confusion matrix, ROC, Information, Entropy.
2. Dependency and uncertainty in training data, linear correlation, mutual information, false neighborhood), anomaly detection, machine learning based anomaly detection.
3. Clustering, low-dimensional representation of high-dimensional data, data compression, (Principal Component Analysis (PCA), SVD, t-SNE, Self Organizing Maps)
4. Communication over a noisy channel (basic types of noisy channels, information conveyed by a channel, the noisy-channel coding theorem)
5. Coding theory, Huffman coding, concepts of arithmetic coding, Hamming codes

Math for Artificial Intelligence and Data Science (guarantor Jan Valdman)

1. Linear algebra: A solution of a system of linear equations by the Gaussian elimination, an explanation of the (reduced) row echelon form and its applications, the rank-nullity theorem.
2. Matrix decompositions: Eigenvectors and Eigenvalues, singular values and singular decomposition, application(s) of SVD.
3. Analytical geometry: Vector (and matrix) norms and their computation, inner products and orthogonality (orthonormality), orthogonal projections, Gram-Schmidt process.
4. Vector Calculus: partial differentiation, gradients, higher-order derivatives, multivariate Taylor series.
5. Optimization: gradient descent, constrained optimization, Lagrange multipliers, Linear programming.



Theoretical Fundamentals of AI (guarantor Peter Jüttner)

*** Before summer semester 2022**

1. Definition of the Semantics of a recursive function based on fixedpoint theory
2. Turing Machine Structure and Programming (Definition)
3. Equivalence of goto – and while programs in terms of computability (proof)
4. Existence of non-computable functions (Proof)
5. O-notation for the definition of complexity classes (Definition)

Theoretical Fundamentals of AI (guarantor Ceazar Ionescu)

*** Summer semester 2022**

1. First-order logic (FOL): syntax of FOL, introduction and elimination rules for connectives and quantifiers, formal proofs in a Fitch system.
2. Semantics: language, reference function, model, syntactic form of sentences, truth-conditions, modalities, possible world semantics, probability and possible worlds.
3. Probability: sample space, event, probability distribution, conditional probabilities, independent events, Bayes Theorem, degrees of belief, Bayesian confirmation theory
4. Information theory: encoding and decoding functions, efficient communication, compression using Huffman trees, Shannon entropy, entropy, joint probability distributions, joint entropy, interdependence, mutual information
5. Decisions and games: utility, expected utility of an action, normal form for static games, dominant strategies, Pareto optimality, Nash equilibria, public goods games.

Theoretical Fundamentals of AI (guarantor Markus Mayer)

*** From summer semester 2023**

1. Language and logic: Syntax of FOL, translation from and to FOL, creating a universe of discourse.
2. Formal first-order logic (FOL) proofs: Introduction and elimination rules for connectives and quantifiers, formal proofs in a Fitch system.
3. Probability: sample space, event, rule of product, combinations and permutations, probability, conditional probabilities, independent events, Bayes



Theorem.

4. Discrete and continuous random variables: Expectation and variance, Bernoulli and normal distribution, Law of large numbers and Central limit theorem, joint probability distributions, marginal distributions, covariance, correlation, and independence.
5. Optimization: On overview of function optimization algorithms (mainly of the Black-Box type) and application hints (e.g. grid search, Monte Carlo search, coordinate descent, ...).

Part on Artificial Intelligence – subjects:

AI & Software Development (guarantor Ceazar Ionescu)

*** Before summer semester 2022**

1. Functional programming in Haskell: expressions and evaluation, types and type classes, list comprehensions, higher-order functions
2. Functional programming in Python: lambda expressions, generators and generator comprehensions, higher-order functions (map, filter, reduce, zip, etc)
3. Symbolic expressions: Haskell types and Python classes for symbolic expressions, evaluation of symbolic expressions in an environment, implementing automatic differentiation in Haskell and Python.
4. SAT and SMT: satisfiability, satisfiability modulo theories, formulating problems in an SAT / SMT framework
5. Z3 SAT/SMT solver in Python: formulating constraints and optimization functions in Z3, solving logical puzzles and combinatorial optimizations.

AI & Software Development (guarantor Ceazar Ionescu)

*** Summer semester 2022**

1. Code generating models: language model, n-gram models, learning n-gram models, word embeddings, generating sentences, byte-pair encodings, GPT and AlphaCode.
2. Logic programming with Prolog: clauses, assertions, queries, rules, unification, backtracking, the cut operator, lists, reversible predicates.
3. Functional programming with Haskell: types and classes, defining functions, list comprehensions, higher-order functions.



4. Monads: collections; dynamical systems: deterministic, stochastic, non-deterministic; trajectories of dynamical systems; generic representation of trajectories using monads.
5. Dependently-typed programming using Idris: dependent types, specification of programs using dependent types, correctness proofs, specification and verification of a simple compiler.

AI & Software Development (guarantor Ceazar Ionescu)

*** From summer semester 2023**

1. Large language models and software. Explain the structure and functioning of a large language model and discuss the use of LLMs for developing software, particularly in relation to program correctness.
2. Prolog. Explain the building blocks of Prolog programs (facts, rules, queries) and the backtracking mechanism. Discuss the relationship between "programming in logic" and correctness, and how Prolog comes up short of the ideal because of features like the cut operator and "negation as failure"
3. Functional programming. Explain the main difference between functional and imperative programming. Give examples of Haskell datatypes, functions, type classes. Explain how to obtain general fold functions for Haskell datatypes.
4. Equational reasoning. Proving correctness from specifications. Explain the idea of equational reasoning and give examples. Discuss why functional languages are more suitable for correctness proofs than imperative ones.
5. Dependently-typed programming: languages for programs and proofs. Explain the propositions-as-types isomorphism and why dependent types are needed for formulating correctness properties. Define the identity type in Agda and formulate some of its properties.

AI & Software Development (guarantor Thomas Ewender)

*** From summer semester 2025**

1. Large language models and software: structure and operation of language models, LLMs, principles of model training, and applications of LLMs in software development.
2. Code-generating models: classical and learned language models, n-gram approaches, word embeddings, sentence generation, byte-pair encoding, transformer architectures, multi-head attention, supervised fine-tuning, and next-token prediction.
3. Functional programming in Haskell: evaluation of expressions, types and type classes, list comprehensions, and higher-order functions.



4. Symbolic expressions: Haskell type representations for symbolic expressions and their evaluation within an environment.
5. Equational reasoning: method of proving correctness from specifications using equality transformations, with examples.

Advanced Machine Learning (guarantor Christina Bauer)

1. Linear and polynomial regression: Explain how the squared error cost function is applied to linear/ polynomial regression and how the gradient descent algorithm is used to solve the optimization objective.
2. Logistic regression: Explain the difference of logistic regression and linear regression in terms of the used hypothesis, cost function, and optimization objective.
3. Neural networks: Describe the model structure of a neural network and how forward and backward propagation is used to solve the optimization objective.
4. Support vector machines: Explain the difference of SVM to logistic regression and how kernels are used in this context.
5. Evaluating learning algorithms: Describe the bias and variance problem and how regularization, cross-validation, and other methods can be used to solve for these problems.

Computational Intelligence (guarantor Ivo Bukovsky)

1. Learning. Describe the difference between unsupervised and supervised learning including differences in training datasets. Explain the principle of the supervised gradient learning, how the gradient is calculated and hyperparameters of the algorithm.
2. Shallow and deep neural networks (DNNs). Typical neurons, their activations and typical activation functions. Describe convolution, pooling and dense layers and their parameters. Application of DNNs for object detection in images.
3. Recurrent neural networks (RNN). Explain what the recurrent connections and necessity of their specific evaluation are. Explain fully recurrent and LSTM neural architecture.
4. Reinforcement learning. Explain the principle. What is the environment state, observation, reward, policy? Describe the actor-critic architecture.
5. Nature inspired optimization. Explain the principle of the genetic algorithm and the particle swarm optimization algorithm.



Part on Data Science – subjects:

Advanced Data Storages and Analyses (guarantor Ivo Bukovsky)

1. Indexing and query execution and optimization in relational and no-SQL databases. Explain the differences between index types (B+tree, hash, bitmap, map/reduce) and the process of query optimization.
2. Database scaling and big data processing. Describe different approaches to database scaling (Vertical and Horizontal scaling, application level, mirroring, partitioning, sharding) and non-database approaches to big data storing and processing (DW, DataLakes, ETL/ELT).
3. CAP theorem. Explain basic paradigms of SQL and no-SQL databases (ACID/BASE, Brewer's diagram) and illustrate how the database type corresponds with the data structure and purpose in specific cases.
4. Data analyses and processing. Explain different data storage and processing approaches (SQL, no-SQL, DW/DataLake, RDF and their influence on data analyses. What type of storage is suitable for what type of tasks? Describe basic data-related tasks (visualization, deep data analyses, reporting, data processing) and discuss the relationship between data storage type and the task.
5. Methods of data analysis. Describe the basic data analysis methods (histogram, boxplot, correlation, pairwise covariance, ...) and interpret their results on a simple example. Describe principles of advanced data analysis methods (clustering, regression, classification) and describe the interpretation of their possible outputs.

Parallel programming and computing (guarantor Milan Predota)

1. MPI parallelization: Commands for collective and non-collective communication.
2. OpenMP parallelization: Principle, commands, variables.
3. Compilation and execution of MPI, OpenMP and OpenCL (GPU) programs. Setting the number of threads, monitoring the efficiency of parallel jobs. Input and output of data.
4. Shared vs. distributed memory, MPI vs. OpenMP. GPU memory. Advantages/disadvantages.
5. Parallelization of cycles – implementations, efficiency.



Distributed Algorithms (guarantor Jan Fesl)

1. Leader Election Algorithms (motivation, examples for logical ring and tree).
2. Time in distributed systems, problem definition (physical vs logical time), Lamport's clock. Christians and Berkeley algorithm.
3. Mutual exclusion algorithms, problem definition, Algorithms Lamport, Ricart-Agrawala.
4. Deadlock detection in distributed systems, problem definition (based on resource allocation graph), Lomet algorithms, Chandy-Misra-Haas algorithm.
5. Remote Procedure Calls, motivation, principles. Google RPC, motivation, types of services, message and methods specification based on Protocol Buffers.

Feature Engineering for Data Science (guarantor Ivo Bukovsky)

*** From academic year 2024 / 2025**

- 1) Feature Engineering (FE) in Data Science and Machine Learning: Definition, importance, role of FE. Features and feature vectors in neural networks. Examples of FE techniques and their properties. Manual and automated feature extraction methods and their applications. Overview of FE methods in prediction, classification, and detection for time series, image processing, and natural language processing.
- 2) FE for Time Series: Time-delay embeddings and feature vectors for time series prediction and analysis. Sliding window methods and time trajectory of time series features. Fundamental and advanced statistical features. Windowed statistical features for time series classification. Rolling window histogram, distribution parameters, Shannon entropy, and vibration analysis. 2-D feature extraction via FFT and spectrograms.
- 3) Principal Component Analysis (PCA) in Feature Engineering: Mathematical principles of PCA. Visualization of multidimensional features in 2-D. Noise suppression and time-trajectory of principal components. PCA integration in machine learning. Applications in feature extraction for images. Comparison of PCA with SVD, UMAP, and SOM for FE.
- 4) Fractal-Based FE: Fractal geometry in FE and self-similar structures in data. Monofractals and multi-fractals, their characteristics, and measurement techniques. Hausdorff dimension and Box-counting dimension. Rényi entropies and generalized fractal dimensions. Applications in time series analysis and image processing. Integration with neural networks and machine learning models. Role in pattern recognition, anomaly detection, and classification. Advantages and limitations.