Additional Admissions Form Master Program in Data Science & Artificial Intelligence

Instructions for filling out this form:

- Please fill out **all** fields within this form, unless indicated otherwise.
- Please feel free to add rows to the tables in this form if necessary.
- Copy the detailed and comprehensive course descriptions of all the relevant courses you have taken and the courses you are currently taking. Copy this from your study guide and **highlight** in it (in yellow) all the key words mentioned in our admission criteria and where you think you meet these requirements. You may add a **brief** explanation if you think the official course description of your university is incomplete, but it should be very clear what is the official course description and what is your own addition. **Copy all information in this form and safe as PDF; We use this form primarily to determine whether you meet the admission requirements so study very carefully all the courses relevant to the master's that you think you meet! This will benefit your assessment!**

Personal details:

Full name (as on passport)	Zeng Qin Tao
TU/e student number	2277522

Information on bachelor's degree:

Name of bachelor's degree program	B.Sc. Computer Science
(Expected) graduation date	2025/06/23
Name of university	University of Birmingham
City and country of university	Birmingham, United Kingdom

Courses showing eligibility for the Master program in Data Science & Artificial Intelligence:

Please list all courses that prove you have passed one or more courses (5 ECTS) on each of the following subjects:

- **Linear Algebra**
  - calculate with matrices and vectors
  - solve linear systems with Gauss(-Jordan) elimination
  - understand and apply rank, orthogonality, (in)dependency, eigenvalue decomposition and eigenvectors
  - implement linear algebra calculations
- **Logic**
  - propositional logic and predicate logic
  - reason with logical formulas
  - provide proofs using various proving techniques (incl. induction, case distinction, contradiction)
- **Discrete Mathematics**
  - concepts and techniques from discrete mathematics (set theory, functions, relations, orderings)
  - knowledge of basic graph theory (graphs, trees, and their properties)
  - use them in computations and using them in formal reasoning
- **Probability and Statistics**
  - introductory probability theory, knowledge of discrete and continuous random variables
  - descriptive statistics including the theory and practice of confidence intervals, hypothesis testing and estimation theory
- **Data structures and Algorithms**
  - algorithm design techniques and use of standard data structures
  - prove correctness of algorithms and reason about algorithm complexity
- **Object-Oriented Programming and Software Development**
  - write programs from scratch in imperative or object-oriented languages
  - use general algorithmic techniques (aggregation, searching, sorting, recursion)
  - apply the principles of code quality and software engineering
- **Data Modelling and Databases**
  - design data models (E-R diagrams, UML) from natural language requirements
  - querying data in relational databases based on natural language requirements

- **Machine learning/Data mining**
  - theoretical foundations of data mining and machine learning
  - apply feature selection and extraction
  - apply supervised learning (classification and regression)
  - apply unsupervised learning (clustering and matrix-factorization)
  - apply evaluation methods and understand overfitting
- **Visualization**
  - Basic principles of visualization
  - design, implement and evaluate visualization tools
  - Implement data transformation, visualizations using visual variables and interaction principles
- **Group project work, that include the skills presenting and writing.**

Required TU/e subject	**Course name, code, and description of course contents (as given in the course syllabus/course catalogue)**	**Credits of the course/total credits of bachelor** *(for example 5 out of 180)* **and final grade**
Linear Algebra	**1. LC Mathematical and Logical Foundations of Computer Science, 06 35324** Mathematical and logical reasoning underpins almost all of Computer Science, from linear algebra in graphics and machine learning, to algebra in cryptography and logic in verification. This module introduces essential topics in mathematics and logic, including linear algebra, abstract algebra, set theory (a fundamental language of mathematics), and propositional and predicate logic. The key ideas will be illustrated with applications across a range of topics in computer science. Learning Outcomes: • Solve mathematical problems in algebra and set theory • Understand and apply algorithms for key problems in logic such as satisfiability. • Write formal proofs for propositional and predicate logic • Apply mathematical and logical techniques to solve a problem within a computer science setting	1. 20 out of 360 ECTS Grade: 60/100 2. 20 out of 360 ECTS Grade: Not Finished Yet

	2. LH Machine Learning, 06 38965 Machine learning studies how computers can autonomously learn from available data, without being explicitly programmed. The module provides a solid foundation in machine learning by giving an overview of the core concepts, theories, methods, and algorithms for learning from data. The emphasis will be on the underlying theoretical foundations, illustrated through a set of methods used in practice. This will provide the student with a good understanding of how, why, and when various machine learning methods work. Learning Outcomes: • Demonstrate knowledge and understanding of core ideas and foundations of automated learning from data. • Demonstrate understanding of broader issues of learning and generalisation in machine learning. • Demonstrate the ability to apply the main approaches to unseen examples. **Self-Reflection:** This course heavily involves linear algebra in various aspects of machine learning. Key operations such as matrix multiplication, dot products, and vector addition are fundamental in algorithms like gradient descent, linear regression, and neural networks. Gauss elimination is utilized to solve optimization problems, enabling efficient solutions in model training. These linear algebra principles form the foundation for understanding and implementing machine learning models effectively.	
Logic	**LC Mathematical and Logical Foundations of Computer Science, 06 35324** Mathematical and logical reasoning underpins almost all of Computer Science, from linear algebra in graphics and machine learning, to algebra in cryptography and logic in verification. This module introduces essential topics in mathematics and logic, including linear algebra, abstract algebra, set theory (a fundamental language of mathematics), and propositional and predicate logic. The key ideas will be illustrated with applications across a range of topics in computer science. Learning Outcomes: • Solve mathematical problems in algebra and set theory. • Understand and apply algorithms for key problems in logic, such as satisfiability. • Write formal proofs for propositional and predicate logic.	20 out of 360 ECTS Grade: 60/100

	• Apply mathematical and logical techniques to solve a problem within a computer science setting.	
Discrete Mathematics	**1. LC Data Structure & Algorithms, 06 30175** Algorithms lie at the heart of Computer Science and software development. They embody the way in which we solve problems using computers. This module introduces the fundamentals of data structures and algorithms. Data structures will be formulated to represent information in such a way that it can be conveniently and efficiently manipulated by the algorithms that are developed. The ideas will be presented both abstractly, and via problem-solving and implementations. Learning Outcomes: • Design and implement data structures and algorithms. • Argue that algorithms are correct and derive time and space complexity measures. • Explain and apply data structures in solving programming problems. • Make informed choices between alternative data structures, algorithms, and implementations, justifying choices on grounds such as computational efficiency. Self-Reflection: This course provides foundational knowledge in data structures and algorithms, with a focus on understanding how to efficiently represent and manipulate data. It offers insights into basic graph theory, including concepts like graphs, trees, and their properties. We also learn how to apply these data structures to solve programming problems effectively. Additionally, the course emphasizes reasoning about the correctness of algorithms, analysing their time and space complexity, and making informed choices between alternative algorithms and data structures based on computational efficiency. **2. LC Mathematical and Logical Foundations of Computer Science, 06 35324** Mathematical and logical reasoning underpins almost all of Computer Science, from linear algebra in graphics and machine learning, to algebra in cryptography and logic in verification. This module introduces essential topics in mathematics and logic, including linear algebra, abstract algebra, set theory (a fundamental language of mathematics), and propositional and	1. 20 out of 360 ECTS Grade: 45/100 2. 20 out of 360 ECTS Grade: 60/100

	predicate logic. The key ideas will be illustrated with applications across a range of topics in computer science. Learning Outcomes: • Solve mathematical problems in algebra and set theory. • Write formal proofs for propositional and predicate logic. • Understand and apply algorithms for key problems in logic, such as satisfiability. • Apply mathematical and logical techniques to solve a problem within a computer science setting.	
Probability and Statistics	**LI Artificial Intelligence 2, 06 34255** Artificial Intelligence (AI) and Machine Learning are often applied in situations characterised by various kinds of uncertainty, for example uncertainty in data measurements, missing data or uncertainty in our prior knowledge about the problem. This module will provide principles that enable AI to treat uncertainty consistently in inference, search, optimisation and learning. Learning Outcomes: • Conceptually understand frameworks for consistent treatment of uncertainty in AI. • Understand principles of inference and fitting to data in AI models under uncertainty. • Describe and understand randomised search and optimisation techniques utilised in AI. Self-Reflection: In this course, we apply concepts from probability and statistics to model and address uncertainty in Artificial Intelligence. The lectures provided a strong foundation in understanding both discrete and continuous random variables and their distributions, which are crucial for AI models that deal with uncertainty in data, measurements, and decision-making. Key topics included probability distributions, which help us reason about relationships between variables in complex AI problems. Additionally, we explored randomised search and optimisation techniques used in AI, which often rely on probabilistic methods to find solutions when facing uncertainty.	20 out of 360 ECTS Grade: 43/100

Data Structures and Algorithms	**LC Data Structure & Algorithms, 06 30175** Algorithms lie at the heart of Computer Science and software development. They embody the way in which we solve problems using computers. This module introduces the fundamentals of data structures and algorithms. Data structures will be formulated to represent information in such a way that it can be conveniently and efficiently manipulated by the algorithms that are developed. The ideas will be presented both abstractly, and via problem-solving and implementations. Learning Outcomes: • Design and implement data structures and algorithms. • Argue that algorithms are correct and derive time and space complexity measures. • Explain and apply data structures in solving programming problems. • Make informed choices between alternative data structures, algorithms, and implementations, justifying choices on grounds such as computational efficiency.	20 out of 360 ECTS Grade: 45/100
Object-Oriented Programming and Software Development	**1. LC Object Oriented Programming, 06 34229** Object-oriented programming is one of the most popular techniques in industrial software development. This module introduces students to the **principles of object-oriented programming**, imperative algorithms, and data structures. Learning Outcomes: • Explain and apply the fundamental constructs of imperative and object-oriented programming, and data structures. • Analyse computer programs, for example, by determining the behaviour of a program from its source code or by completing and/or correcting partially written programs. • Write, test, and debug computer programs, where appropriate making effective use of an integrated development environment (IDE) and other programming aids. • Design and document complete computer programs to solve given software problems, including some open-ended tasks.	1. 20 out of 360 ECTS Grade: 91/100 2. 20 out of 360 ECTS Grade: 76/100

	2. LI Software Engineering, 06 40098 This module introduces students to the field of software engineering and the principles of systematically engineering large-scale software systems. The module covers widely used techniques for engineering requirements, designing, and modelling, and architecting dependable and evolvable software. The module discusses Software Quality Assurance, testing, and project management with an appreciation of the economical, legal, and ethical aspects. Students will apply their software engineering knowledge in the development of a team project. Learning Outcomes: • Describe and apply techniques for systematically engineering requirements, designing, architecting, and testing software systems, along with the technical, economical, legal, and ethical trade-offs involved. • Describe and apply the basic principles of software project management, metrication, and quality assurance. • Understand advances in industrial software engineering. • Demonstrate teamwork and leadership in developing a group project using good software engineering practices.	
Data Modelling and Databases	**1. LC Full Stack Application Development, 06 34252** Many modern software applications involve a full software stack – a client-side front-end, a server to interpret and process client requests, and a database to store the application's content. This module will introduce a range of tools and techniques for full stack software development. Learning Outcomes: • Design, populate, and query a database. • Design and implement a front-end interface. • Design and implement server-side application software. • Integrate a database, server, and front-end into a full software stack. • Use modern development tools effectively.	1. 20 out of 360 ECTS Grade: 62/100 2. 20 out of 360 ECTS Grade: 84/100 3. 20 out of 360 ECTS Grade: 76/100

2. LI Security and Networks, 06 30195

As computers are embedded in everyday life, protection against the criminal or unauthorized use of electronic data is essential and measures must be taken to achieve this. The module will introduce a range of topics in the theory and practice of computer security, including attacks, vulnerabilities, and defences. Cloud and Web services will be used as a motivating example for the importance of security.

Learning Outcomes:

- Understand basic concepts of cryptography and SQL.
- Understand basic concepts of cloud services, in particular storage.
- Demonstrate an understanding of the threats to data stored on a computer, locally or in the cloud.
- Demonstrate an understanding of the threats to data sent on the network.
- Identify risks and use techniques to eliminate or mitigate them.

3. LI Software Engineering, 06 40098

This module introduces students to the field of software engineering and the principles of systematically engineering large-scale software systems. The module covers widely used techniques for engineering requirements, designing, and modelling, and architecting dependable and evolvable software. The module discusses Software Quality Assurance, testing, and project management with an appreciation of the economical, legal, and ethical aspects. Students will apply their software engineering knowledge in the development of a team project.
Learning Outcomes:

- Describe and apply techniques for systematically engineering requirements, designing, architecting, and testing software systems, along with the technical, economical, legal, and ethical trade-offs involved.
- Describe and apply the basic principles of software project management, metrication, and quality assurance.
- Understand advances in industrial software engineering.

	• Demonstrate teamwork and leadership in developing a group project using good software engineering practices. Self-Reflection: In this module, we implemented UML diagrams to design and model large-scale software systems. The use of UML diagrams, such as use case diagrams, class diagrams, and sequence diagrams, allowed us to visualize and document system requirements, architecture, and interactions systematically.	
Machine Learning/Data Mining	**1. LC Artificial Intelligence 1, 06 34238** Artificial Intelligence (AI) is the area of Computer Science which studies algorithms capable of problem solving and learning. In recent years, AI systems have become increasingly prominent in society and industry. This module will introduce fundamental concepts from AI and Machine Learning, covering knowledge representation, search, optimisation and learning. It will provide experience applying these concepts to solve practical problems. Learning Outcomes: • Demonstrate an understanding of traditional AI approaches. • Demonstrate an understanding of the core principles of Optimisation and Machine Learning. • Demonstrate an understanding of the relationship between basic concepts of univariate differentiation and techniques of AI. • Apply core principles of artificial intelligence to solve problems. **2. LI Artificial Intelligence 2, 06 34255** Artificial Intelligence (AI) and Machine Learning are often applied in situations characterised by various kinds of uncertainty, for example uncertainty in data measurements, missing data or uncertainty in our prior knowledge about the problem. This module will provide principles that enable AI to treat uncertainty consistently in inference, search, optimisation and learning. Learning Outcomes: • Conceptually understand frameworks for consistent treatment of uncertainty in AI. • Understand principles of inference and fitting to data in AI models under uncertainty.	1. 20 out of 360 ECTS Grade: 52/100 2. 20 out of 360 ECTS Grade: 43/100 3. 20 out of 360 ECTS Grade: Not Finished Yet 4. 20 out of 360 ECTS Grade: Not Finished Yet 5. 20 out of 360 ECTS Grade: Not Finished Yet

- Describe and understand randomised search and optimisation techniques utilised in AI.

3. LH Machine Learning, 06 38965

Machine learning studies how computers can autonomously learn from available data, without being explicitly programmed. The module provides a solid foundation in machine learning by giving an overview of the core concepts, theories, methods, and algorithms for learning from data. The emphasis will be on the underlying theoretical foundations, illustrated through a set of methods used in practice. This will provide the student with a good understanding of how, why, and when various machine learning methods work.

Learning Outcomes:

- Demonstrate knowledge and understanding of core ideas and foundations of automated learning from data.
- Demonstrate understanding of broader issues of learning and generalisation in machine learning.
- Demonstrate the ability to apply the main approaches to unseen examples.

Self-Reflection:
This module provided hands-on experience in applying supervised learning, such as classification and regressions. Overfitting was addressed by implementing techniques like cross-validation and regularization (L1 and L2 norms) during model evaluation.

4. LH Natural Language Processing, 06 37810

Natural Language Processing (NLP) enables computers to understand and reason about human languages such as English and has resulted in many exciting technologies such as conversational assistants, machine translation, and intelligent internet search. This module provides the theoretical foundations of NLP as well as applied techniques for extracting and reasoning about information from text.

Learning Outcomes:

- Demonstrate an understanding of the major topics in Natural Language Processing.
- Understand the role of machine learning techniques in widening the coverage of NLP systems.

	• Demonstrate an ability to apply knowledge-based and statistical techniques to real-world NLP problems. **5. LH Neural Computation, 06 32167** This module introduces the basic concepts and techniques of neural computation, and its relation to automated learning in computing machines. It covers the main types of formal neuron and their relation to neurobiology, showing how to construct large neural networks and study their learning and generalization abilities in the context of practical applications. It also provides practical experience of designing and implementing a neural network for a real-world application. Learning Outcomes: • Understand the relationship between real brains and simple artificial neural network models. • Describe and explain some of the principal architectures and learning algorithms of neural computation. • Explain the learning and generalisation aspects of neural computation. • Demonstrate an understanding of the benefits and limitations of neural-based learning techniques in context of other state-of-the-art methods of automated learning. • Apply neural computation algorithms to specific technical and scientific problems.	
Visualization	**LH Computer Science Project, 06 26581** During the course, students will have developed a passion for topics in computer science, such as artificial intelligence or computer security, and the project is the ideal opportunity to explore one of those topics in depth. Students will take a complete project through the concept, design, implementation, and evaluation phases, gaining crucial experience that can be transferred to a commercial or research environment. The project normally includes writing a substantial piece of software or developing some hardware. Occasionally, it can consist of conducting research other than by writing software. Learning Outcomes: • Carry out a substantial computer science problem-solving task.	40 out of 360 ECTS Grade: Not Finished Yet

	• Work independently and prioritise different components of the work; manage a large project effectively. • Take decisions and justify them convincingly. • Orally present work undertaken, and answer questions about it convincingly. • Write a formal report, detailing work undertaken and conclusions reached. Self-Reflection: In thesis project, I designed and implemented an interactive dashboard to showcase the skills of data visualization. The dashboard provides real-time insights through charts, graphs, and interactive filters. Through this project, I developed a strong understanding of the principles of data visualization, including data preprocessing, chart selection, and creating user-friendly designs	
Group project work	**1. LI Software Engineering, 06 40098** This module introduces students to the field of software engineering and the principles of systematically engineering large-scale software systems. The module covers widely used techniques for engineering requirements, designing, and modelling, and architecting dependable and evolvable software. The module discusses Software Quality Assurance, testing, and project management with an appreciation of the economical, legal, and ethical aspects. Students will apply their software engineering knowledge in the development of a team project. Learning Outcomes: • Describe and apply techniques for systematically engineering requirements, designing, architecting, and testing software systems, along with the technical, economical, legal, and ethical trade-offs involved. • Describe and apply the basic principles of software project management, metrication, and quality assurance. • Understand advances in industrial software engineering. • Demonstrate teamwork and leadership in developing a group project using good software engineering practices. **2. LI Team Project, 06 26263** Large software projects require the combined effort of a team of skilled developers. In this module, students will learn the basics of teamwork in the context of developing a large	1. 20 out of 360 ECTS Grade: 76/100 2. 20 out of 360 ECTS Grade: 72/100 3. 20 out of 360 ECTS Grade: Not Finished Yet 4. 40 out of 360 ECTS Grade: Not Finished Yet

	programming project. Students work in teams of about six people to design, implement, and document the code. Students are expected to observe good software engineering practices and work effectively in a team. Students will use industry-standard approaches such as continuous integration. Learning Outcomes: • Design and implement a large software project. • Use good software engineering practices in a project. • Demonstrate teamwork and leadership. **3. LH Human-Computer Interaction, 06 30214** Human-computer interaction (HCI) explores the technical and psychological issues arising from the interface between people and machines. Understanding HCI is essential for designing effective hardware and software user interfaces. This module teaches the theory and practice of HCI methodologies for both design and evaluation. Learning Outcomes: • Explain and discuss the key capabilities and limitations in human cognitive performance and relate this to the design of HCI systems. • Demonstrate an understanding of the use of cognitive modelling techniques in HCI. • Select and evaluate appropriate HCI design methodologies for real-world design problems, particularly software. • Demonstrate an understanding of the scope and importance of HCI systems across a range of application domains. Self-Reflection: In this module, we demonstrated strong teamwork as a group of five while collaboratively writing and documenting a comprehensive project report. This report covered the entire design process, from user research to prototype development and evaluation. We applied key HCI principles, such as usability heuristics, user-cantered design to address real-world design challenges. Additionally, we employed appropriate evaluation methods, such as usability testing, to refine our designs.	

	4. LH Computer Science Project, 06 26581 During the course, students will have developed a passion for topics in computer science, such as artificial intelligence or computer security, and the project is the ideal opportunity to explore one of those topics in depth. Students will take a complete project through the concept, design, implementation, and evaluation phases, gaining crucial experience that can be transferred to a commercial or research environment. The project normally includes writing a substantial piece of software or developing some hardware. Occasionally, it can consist of conducting research other than by writing software. Learning Outcomes: • Carry out a substantial computer science problem-solving task. • Work independently and prioritise different components of the work; manage a large project effectively. • Take decisions and justify them convincingly. • Orally present work undertaken, and answer questions about it convincingly. • Write a formal report, detailing work undertaken and conclusions reached.	
Other Modules	**1. LI Functional Programming, 06 34253** This module develops practical programming skills in a typed functional programming language. It will strengthen the algorithmic and design skills of the students within the functional framework, and will introduce them to some advanced programming language features. Learning Outcomes: • Present the basic ideas of functional programming language. • Demonstrate the main elements of good functional programming style. • Illustrate some of the uses and applications of functional programming. • Understand and apply more advanced features of typed functional programming. **2. LI Operating Systems and Systems Programming, 06 38059** An Operating System is the system software that manages computer hardware, hardware and software resources, and provides common services for user programs. System programming is	1. 20 out of 360 ECTS Grade: 71/100 2. 20 out of 360 ECTS Grade: 46/100 3. 20 out of 360 ECTS Grade: 53/100

	the type of programming necessary to produce software, such as operating systems, that deal with hardware, provide services to other software or manage performance constraints. This module teaches the technology of operating systems and introduces students to the challenges of systems-level programming. Learning Outcomes: • Demonstrate understanding of computer architecture and operating systems. • Program with pointers and memory management. • Write code that interacts with the OS kernel. **3. LC Theories of Computation, 06 35393** Computers have been used to solve an astonishing range of different problems, but this does not mean that they can be used to solve all possible problems: some cannot be solved efficiently, and some cannot be solved at all. In this module, we will introduce a set of principles and techniques for formalising computation and computability to understand what problems can be solved, how efficiently they can be solved, and what problems cannot be solved. We will develop mathematical models of computations using ideas such as automata theory (including Turing machines), of formal languages using ideas such as regular expressions and grammars and will conclude by considering the notions of non-computability and complexity. Learning Outcomes: • Explain and apply mathematical models of computations. • Describe and use the connection between finite automata and regular language. • Explain and apply concepts from automata theory, formal language theory, computability theory and complexity theory • Explain non-computability and undecidability issues	