Department of
Civil & Environmental Engineering
THE UNIVERSITY OF UTAH

CVEEN 5565

Machine Learning for Civil Computing

Course Handouts



Chenxi Liu

University of Utah

Spring 2026

Department of Civil & Environmental Engineering
THE UNIVERSITY OF UTAH

Course Syllabus

Class:	CVEEN 5565 (001): Machine Learning for Civil Computing	
Time:	Lecture sessions: 3:00 PM – 4:20 PM on Mondays and Wednesdays	
Location:	LCB 121	
Website:	Canvas Link	
Registration:	Please send an email to Mina (Mina.Castro@utah.edu) for registration.	
Objective & Outcomes	The objective of this course is to provide students with a solid introduction to the principles of deep learning as applied to civil engineering, with a focus on computational techniques and their practical applications. Through lectures, discussions on real-world applications, and data-driven projects, students are expected to obtain a general understanding of deep learning principles and the state-of-the-art in computational methods. In addition, the course is designed to build a solid foundation for students to explore interdisciplinary research, integrating machine learning with civil engineering to tackle complex challenges. With this knowledge, students will be equipped to contribute effectively to both academic research and practical engineering projects and engage in cross-disciplinary efforts to address pressing civil engineering issues using ML/AI tools. By the end of the course, students will gain a strong grasp of core deep learning concepts, including key architectures such as fully connected neural networks, convolutional neural networks, and recurrent neural networks, and will be able to apply these techniques to analyze large-scale civil engineering datasets. This preparation will enable them to design and implement machine learning models to solve practical engineering problems, improving their skills in model selection, training, validation, and interpretation. Students will also be expected to evaluate the limitations and risks of the models they build, including error asymmetry in safety-critical prediction, bias and distribution shift in training data, and the data governance implications of deployed sensing systems.	
Description:	This course on Deep Learning for Civil Computing is crafted to bridge the significant gap between advanced computational methods and their practical applications in civil engineering. The curriculum offers an introductory exploration of essential machine learning principles and a range of deep learning models, including neural networks, reinforcement learning, and generative systems. With a strong emphasis on practical application, the course delves into how these cutting-edge technologies can be effectively utilized in critical civil engineering domains such as traffic management, infrastructure resilience, and sustainable development. Through a combination of lectures and the projects with practical applications, students will develop a foundational understanding of how to implement these ML/AI tools to solve real-world challenges in civil engineering. This course prepares them to integrate and leverage sophisticated computational techniques in both academic and practical civil engineering projects.	
Instructor:	Chenxi Liu	
	Office: Office hours:	2032 MCE 2:00 PM - 3:00 PM every Mon and Wed, or by appointment

	Phone:	(801) 585-0476 E-mail: chenxi.liu@utah.edu
Grading Policy:	Individual Assignments: 30%; In-Class Mini-Exams: 30%; Paper Review: 10%; Group Project: 30%	

Department of Civil & Environmental Engineering
THE UNIVERSITY OF UTAH

CVEEN 5565: Machine Learning for Civil Computing

Course Schedule – Spring 2026

Week	Day	Date	Topics	Note
1	M	Jan. 5	Lecture 1: Course Organization and Structure, Introduction and Background	Course Survey Out
	W	Jan. 7	Lecture 2: Basics of Python (I): Jupyter Notebooks, Google Colab, Basic Syntax, and Numpy	Paper Selection Assignment #1 Out
2	M	Jan. 12	Lecture 2: Basics of Python (II): Advance Python Skills	E-Learning (TRB Trip) Project Group Up
	W	Jan. 14	Programming Practice 1: Python Skills Practice	E-Learning (TRB Trip)
3	M	Jan. 19	***Martin Luther King Day (No Class)***	
	W	Jan. 21	Lecture 3: Linear Regression (I): Loss Function, and Gradient Descent	
4	M	Jan. 26	Lecture 3: Linear Regression (II): Regularization	
	W	Jan. 28	Lecture 4: Logistic Regression (I): Classification Problems and Decision Boundaries	
5	M	Feb. 2	Lecture 4: Logistic Regression (II): Loss Function and Gradient Descent	
	W	Feb. 4	Programming Practice 2: House Price Prediction and Structure Failure Prediction	
6	M	Feb. 9	Lecture 5: Artificial Neural Network	
	W	Feb. 11	Assignment 1 Overview and Mini-Exam 1	Assignment # 1 Due Assignment #2 Out **Mini-Exam # 1**
7	M	Feb. 16	***Presidents Day (No Class)***	
	W	Feb. 18	Programming Practice 3: Artificial Neural Network (ANN) for Road Surface Condition Classification	
8	M	Feb. 23	Lecture 6: Basics of PyTorch Programming Practice 4: PyTorch Tutorial	
	W	Feb. 25	Lecture 7: Computer Vision Basics Programming Practice 5: Computer Vision Basics	
9	M	Mar. 2	Lecture 7: Convolutional Neural Network	
	W	Mar. 4	Lecture 8: Object Detection: ResNet, SSD, and YOLO Programming Practice 6: Handwriting Recognition and	

			Object Detection	
10	M	Mar. 9	***Spring Break (No Class)***	
	W	Mar. 11	***Spring Break (No Class)***	
11	M	Mar. 16	**Final Project Idea Presentation and Progress Checkpoint**	Assignment #2 Due **Mini-Exam #2**
	W	Mar. 18	Lecture 9: Recurrent Neural Network (I)	Assignment #3 Out
12	M	Mar. 23	Lecture 9: Recurrent Neural Network (II) Programming Practice 7: Traffic Flow Prediction	
	W	Mar. 25	Lecture 10: Edge Artificial Intelligence (I)	
13	M	Mar. 30	Lecture 10: Edge Artificial Intelligence (II)	
	W	Apr. 1	Programming Practice 8: Model Compression and Edge AI	
14	M	Apr. 6	Lecture 11: Advanced AI Technologies Overview, Responsible AI in Civil Engineering Practice, and Future of ML/AI in Civil Engineering	
	W	Apr. 8	**Paper Review Presentation**	Paper Review Due
15	M	Apr. 13	Guest Lecture: TBD	
	W	Apr. 15	Course Review and Final Project Discussion	Assignment #3 Due **Mini-Exam #3**
16	M	Apr. 20	**Final Project Presentation**	Group Project Due

Note: *The syllabus is not a binding legal contract. The instructor may modify it when the student is given reasonable notice of the modification.*

CVEEN 5565 Course Expectations and Policy

Expectations for Course Work

Assigned Reading

Most lectures have assigned readings (please see the course schedule for details) that you need to finish before attending the classes. Though these assigned readings will not be directly evaluated, it will be greatly beneficial to complete reading them on time because they provide important information for you to better understand the class contents.

Assignments

There will be three assignments, which are **individual efforts** as described in *CVEEN Homework Assignments: Process of Solution and Formatting Requirements* (effective 9/1/2004)*:*

"The completed homework assignments that you turn in for credit must be substantially your own work. It is permissible to discuss the basic concepts and how to solve the problem in a general sense with others prior to working on the assignment. Once you have started a problem, you may ask questions of other students, but the questions should be limited to specific aspects of a problem that you do not understand. It is not acceptable to work on the assignments with another person or in a group where the assignments are worked entirely together. You may get as much help from the Teaching Assistant and Professor for the class as they can legitimately give you during their regularly scheduled office hours or via e-mail (if the Teaching Assistant or Professor is willing to communicate via e-mail). It is not permissible to use either solution manuals or solutions from past classes for homework assignments that are turned in for credit. All assignments must have the following signed pledge at the front of the assignment:

On my honor as a student at the University of Utah, I have neither given nor received unauthorized aid on this assignment.

If the pledge is missing or is not signed, the assignment will not be graded."

Assignments and projects are due by 3:00 PM on the due date. Assignment deliverables must follow the Departmental formatting and style requirements given in *CVEEN Homework Assignments: Process of Solution and Formatting Requirements.*

The course schedule shows approximate dates for assignments and projects. Exact assignment and due dates will be identified during the semester by the instructor and announced during the regularly scheduled classroom meetings

Programming Practices

Throughout this course, students will complete seven hands-on programming practice tasks, provided in Jupyter notebook format (".ipynb"). These tasks are designed to deepen their understanding of the course material and enable them to apply key concepts to practical problem-solving scenarios. Each task involves implementing functions or algorithms that directly relate to lecture content, reinforcing theoretical

knowledge through coding practice. Students are encouraged to start and work on these tasks during the practice session, where they can ask questions and receive immediate feedback. These programming practice tasks are optional and will not contribute to the final course grade. Students have the flexibility to decide whether or not to complete them based on their own learning goals and schedules. However, engaging with these tasks is highly recommended as they provide an excellent opportunity to reinforce learning and build practical coding skills in a supportive environment.

Mini-Exams

There will be three 60-minute mini-exams throughout the course. These exams are open-book and open-notes, allowing students to refer to their own materials during the exam. It is important for students to respect their peers by keeping their materials to themselves and maintaining a quiet, focused environment. **The use of the internet and AI tools (such as ChatGPT, Llama, Gemini, etc.) is strictly prohibited during the mini-exams.** AI tools are permitted elsewhere in this course, as set out under Responsible AI in This Course and Use of AI Tools below; the mini-exams are the component in which each student's own conceptual understanding is measured directly.

Each mini-exam will assess students' understanding of the key concepts covered in lectures and assignments. The content of each exam will build progressively. For instance, Mini-Exam #1 will focus on topics covering from Week 1 to Week 5 including basics of machine learning, linear regression, and logistic regression. Subsequent exams will evaluate material introduced from the previous mini-exam up to the most recent lecture. The purpose of these exams is to gauge students' mastery of foundational principles and their ability to apply them to problem-solving within the course.

Paper Review

The Paper Review component of this course is designed to cultivate students' skills in conducting independent literature reviews on advanced topics in deep learning. A curated list of seminal and state-of-the-art papers will be provided, covering foundational theories and emerging developments across relevant domains. Students will select a paper aligned with their academic or professional interests and perform a comprehensive literature review. This process aims to enhance their ability to identify influential research, critically evaluate the progression of specific topics, and synthesize key contributions along with identifying unresolved challenges.

At the culmination of the semester, students will present their findings, including any associated coding outcomes if applicable, in a professional format that summarizes the paper's contributions, contextual insights, and their own critical analysis. This activity is designed to foster independent research capabilities, promote critical thinking, and develop effective communication skills through peer discussions and presentations. The Paper Review serves as an integral part of the course, bridging theoretical knowledge and research applications while preparing students for scholarly or industry-related pursuits in deep learning.

Group Project

This course includes a group project, offering you the opportunity to apply the concepts learned to a problem of your choice. You are encouraged to select a topic that interests you. Projects typically fall into two categories:

Applications: If you have a specific background or interest, you are encouraged to apply deep neural networks to real-world problems related to your domain of expertise.

Models: You can create a new model or algorithm using deep neural networks, or modify an existing model, applying it to areas such as computer vision, natural language processing, information retrieval, or databases. This track can be more demanding but may lead to publishable work.

Projects require careful planning, well-grounded assumptions, and clear communication of results. Each team will consist of 2-3 members (varies by the course enrollment), to be selected by the students. By Week 2, all students must have joined a team. If any student has not done so, the instructor or TA will assign them to a team. By Week 11, each team will present their project idea, covering research questions, literature review, data sources, and their proposed plan moving forward. By Week 16, teams will deliver their final presentations and submit a project report, including the accompanying code.

At the end of the semester, a peer evaluation will be conducted, where team members will allocate 100 points among their peers based on individual contributions. The individual grade for the project will reflect both the overall team grade and personal contributions, meaning that an individual's final grade could differ from the team's overall score.

A project report should be typed and submitted by each team. It should be written in clear English, contain at least eight sections:

- **Introduction**: Provide a concise overview of the project's objectives and scope.
- **Research Questions**: Clearly define the research questions or problems the project aims to address.
- **Data Description**: Describe the data used in the project, highlighting its key features and characteristics.
- **Methodology**: Outline the model structure in detail and explain how it is applied to the data.
- **Case Study/Experiment**: Provide at least one example of the method or process your program follows.
- **Result Analysis**: Present the findings with a thorough analysis and interpretation of the results.
- **Limitations and Responsible Use**: State the conditions under which your model should not be relied upon, identify sources of bias, coverage gaps, or distribution shift in the data, and describe what additional verification a practicing engineer would need before using your results to support a real infrastructure decision.
- **Conclusion and Future Work**: Summarize the main conclusions and propose potential improvements or applications for future work.

College of Engineering Guideline

The College of Engineering Semester Guidelines contain important dates regarding adding, dropping and withdrawing from classes as well as the College Policy regarding repeating courses.

Please refer to the details here: https://www.coe.utah.edu/students/current/semester-guidelines/

Accessibility

The University of Utah seeks to provide equal access to its programs, services and activities for people with disabilities. If you will need accommodations in the class, reasonable prior notice needs to be given to the Center for Disability Services, 162 Union Building, 581-5020 (V/TDD). CDS will work with you and the instructor to make arrangements for accommodations. (www.hr.utah.edu/oeo/ada/guide/faculty/)

Professional Responsibilities

Students should read the "Student Code" (Policy 6-400: Code of Student Rights and Responsibilities) carefully and know they are responsible for the content. The Policy is located at the following link: http://www.regulations.utah.edu/academics/6-400.html

All students are expected to maintain professional behavior in the classroom setting.[1] Students have specific rights in the classroom, detailed in Section II of the Student Code. The Student Code also specifies academic misconduct, including, but not limited to, cheating, misrepresenting one's work, inappropriately collaborating, plagiarism, and fabrication or falsification of information. The process for resolving any violations of the Student Code, including academic sanctions, is also described.

[1] Discussion threads, e-mails, and chat rooms are all considered to be equivalent to classrooms, and student behavior within those environments shall conform to the Code.

Responsible AI in This Course

The models built in this course are intended for civil infrastructure applications in which prediction error carries public safety, financial, and equity consequences. Responsible AI is therefore treated here as an engineering requirement embedded in the technical material rather than as a separate ethics topic, and it is addressed in three ways.

Responsible model development and evaluation. A model reported as accurate is not the same as a model that can be relied upon. Regularization and the training, validation, and test discipline introduced in Lecture 3 are presented as safeguards against models that generalize poorly once deployed, and threshold and decision boundary selection in Lecture 4 is used to introduce asymmetric error costs. This is made concrete in the structural failure prediction exercise (Programming Practice 2), in which a missed failure and a false alarm are not equivalent outcomes and students are expected to reason about class-specific error rates rather than aggregate accuracy, and in the road surface condition classification exercise (Programming Practice 3), which is used to examine the gap between the conditions represented in a training set and the conditions a deployed model will actually encounter.

Data governance and responsible deployment. The Edge Artificial Intelligence unit (Lecture 10 and Programming Practice 8) presents on-device inference as a privacy-preserving design decision for infrastructure and traffic sensing, since transmitting raw imagery of public rights of way raises surveillance and data governance concerns. Students consider what a deployed civil infrastructure system should compute locally and what it should retain or transmit. In the survey of advanced AI technologies (Lecture 11), the limits of large language and generative models, including output that is fluent but factually wrong, are addressed directly so that model fluency is not mistaken for correctness.

Accountable use of AI tools. Generative AI tools are permitted in the assignments, programming practices, paper review, and group project, and their use must be disclosed, as set out under Use of AI Tools below. Individual conceptual mastery is verified separately through the three closed-book mini-exams, which account for 30 percent of the course grade and in which AI tools and internet access are not permitted. The course therefore distinguishes work in which tool assistance is appropriate and expected from assessment in which the student's own understanding is what is being measured. The group project report requires a Limitations and Responsible Use section, and the paper review requires students to evaluate published work critically and identify unresolved limitations rather than summarize claims.

Course Policies

Class Attendance.

You are encouraged to attend all lectures and practice sessions for this course. However, attendance is not mandatory and will not be graded.

Academic Integrity Policy

You are expected to uphold the highest standards of academic honesty in all your work. **Any form of academic misconduct, such as cheating, copying, or plagiarism, will not be tolerated**. Misconduct includes, but is not limited to, the following:

- Copying from another student's work or allowing your work to be copied.
- Submitting someone else's work as your own, or using AI tools or external resources in a manner not permitted under the Use of AI Tools policy below.

- Using unapproved materials during an exam or assignment.

Consequences for Violations:

- First Incident: If you are found to have engaged in academic misconduct on any assignment, practice session, mini-exam, or project, you will receive a zero for that specific task.
- Second Incident: Any subsequent incident of academic misconduct will result in a failing grade for the entire course.

These incidents will be reported to the Office of Student Conduct, which may lead to further actions according to university policies.

Use of AI Tools

Generative AI tools may be used in the assignments, programming practices, paper review, and group project. Professional engineering practice increasingly involves these tools, and the aim of this course is to develop competent and accountable use rather than to prohibit use. Two conditions apply. First, you remain fully responsible for everything you submit. You must be able to explain any code, derivation, or argument in your submission, and an error introduced by an AI tool is treated exactly as an error you made yourself. Second, submitted assignments and project reports must include a brief statement of which AI tools were used and for what purpose, for example code debugging, literature search, or draft editing. AI tools and internet access are not permitted during the three mini-exams, which are the component of the course in which individual conceptual mastery is assessed directly.

Late Assignment (or Practice) Turn-ins.

You are required to turn in your assignments and programming practice on time. Late turn-ins can be accepted within seven days after the due date but will be discounted at 10% per day it is late. For example, if an assignment is due on Feb. 10, and you turn it in on Feb. 12, it would amount to a deduction of 20% for this two-day late assignment. If you turn in on Feb. 18 or after, it will not be accepted, and no credit will be given to this assignment of more than seven days late.

Late Project Turn-ins.

Project reports and code must be submitted on time. Late submissions for project reports or code will not be accepted, and no credit will be given for any part of the project submitted after the due date.

Final Grade Determination.

Your final grade will be determined by combining your scores on individual assignments, in-class mini-exams, programming practices, paper review, and projects using the following weights:

- Individual Assignments: 30%
- In-Class Mini-Exams: 30%
- Paper Review: 10%
- Group Project: 30%

Letter	Scoring	Qualitative Description
A	100% - 92.5%	No to minor mistakes.
A-	92.49% - 90%	

B+	89.99% - 87.5%	
B	87.49% - 82.5%	
B-	82.49% - 80%	Several minor mistakes; almost no conceptual mistakes.
C+	79.99% - 77.5%	
C	77.49% - 72.5%	
C-	72.49% - 70%	Several mistakes, some major; conceptual mistakes.
D+	69.99% - 67.5%	
D	67.49% - 62.5%	
D-	62.49% - 60%	Many significant mistakes and conceptual errors.
E	59.99% - 0	Non-response or completely incorrect response.

Paper Reading List

Basics of Python

1. Introductory Tutorials

- Tutorial for Absolute Beginners

  A full-length video tutorial ideal for students new to Python. It introduces syntax, data types, loops, and functions in a clear, progressive format: https://www.youtube.com/watch?v=kqtD5dpn9C8

- Google's Python Class

  A free set of written materials and recorded lectures for learners with prior programming experience. It emphasizes problem-solving, debugging, and structured exercises:

  https://developers.google.com/edu/python

2. Advanced Concepts

- Advanced Python Techniques for Efficient Data Analysis

  A practical article demonstrating how to optimize Python workflows using libraries like Pandas and Dask. It also covers data-handling performance and scalable analytics:

  https://python.plainenglish.io/advanced-python-techniques-for-efficient-data-analysis-db6b4d783d3a

Linear Regression

1. Fundamentals

- Linear Regression: Gradient Descent | Google Developers

  Explains the core mathematical principle of minimizing the Mean Squared Error (MSE) loss function using gradient descent. An excellent starting point for model training concepts:

  https://developers.google.com/machine-learning/crash-course/linear-regression/gradient-descent

2. Regularization and Extensions

- Lecture 20 – Ridge Regression and Regularization Methods

  A video lecture explaining ridge (ℓ_2) and LASSO (ℓ_1) regression, showing how regularization mitigates overfitting by constraining model coefficients: https://www.youtube.com/watch?v=SotdlGgsco8

- L1 and L2 Regularization in Machine Learning: Easy Explanation

  An accessible introduction to how ℓ_1 and ℓ_2 penalties affect weight magnitude and improve generalization in regression models:

  https://www.youtube.com/watch?v=pJ5c_uLeg2A

3. Foundational Papers

- "A Rank-Invariant Method of Linear and Polynomial Regression Analysis"

One of the earliest formal treatments of linear regression estimation, introducing the concept of rank-invariance and forming a basis for robust regression techniques:

https://ir.cwi.nl/pub/18446/18446A.pdf

- "Ridge Regression: Biased Estimation for Nonorthogonal Problems"

  The seminal work introducing ridge regression, demonstrating how adding a small bias term can reduce variance and improve predictive performance:

  https://www.jstor.org/stable/pdf/1267351

4. Civil Engineering Applications Papers

- "Use of Local Linear Regression Model for Short-Term Traffic Forecasting"

  Applies linear regression to traffic-flow prediction, highlighting short-term forecasting accuracy and interpretability in transportation engineering:

  https://journals.sagepub.com/doi/epdf/10.3141/1836-18

- "Earthmoving Productivity Estimation Using Linear Regression Techniques"

  Demonstrates regression-based prediction for construction productivity, showing how site variables influence operational efficiency:

  https://ascelibrary.org/doi/epdf/10.1061/%28ASCE%290733-9364%281999%29125%3A3%28133%29

5. Theoretical Perspective Papers

- "Theory of Deep Learning III: Explaining the Non-Overfitting Puzzle"

  Analyzes loss landscapes, gradient descent behavior, and implicit regularization — connecting linear regression foundations to deep-learning theory:

  https://arxiv.org/pdf/1801.00173

Logistic Regression

1. Classification and Decision Boundaries

- Learn Decision Boundary | Logistic Regression

  A visual, hands-on tutorial illustrating how logistic regression separates classes through probabilistic boundaries:

  https://codefinity.com/courses/v2/b71ff7ac-3932-41d2-a4d8-060e24b00129/8957b48a-f0bc-4e88-8645-ce0c019908fb/6ba74933-6cc1-4329-932e-7bad9d515ae8

- Logistic Regression and Decision Boundary

  Conceptual explanation of sigmoid transformation, decision thresholds, and visual examples of class regions in 2D feature spaces:

  https://medium.com/data-science/logistic-regression-and-decision-boundary-eab6e00c1e8

2. Foundational Papers

- "The Origins of Logistic Regression"

  Provides a historical overview of how the logistic model developed from early statistical ideas into a central machine-learning method:

  https://www.econstor.eu/bitstream/10419/86100/1/02119.pdf

- "Logistic Regression: A Brief Primer"

  A concise and intuitive introduction to logistic regression, focusing on model interpretation, odds ratios, and application across disciplines:

  https://onlinelibrary.wiley.com/doi/full/10.1111/j.1553-2712.2011.01185.x

- "A Modern Maximum-Likelihood Theory for High-Dimensional Logistic Regression Models"

  A theoretical exploration of logistic regression behavior under high-dimensional data conditions, addressing convergence and generalization:

  https://www.pnas.org/doi/pdf/10.1073/pnas.1810420116

Artificial Neural Networks (ANNs)

1. Foundational Concepts

Neural Inspiration: Early theoretical work attempted to model the brain's information processing. McCulloch and Pitts (1943) introduced a simplified mathematical model of the neuron, demonstrating how logic could be implemented using neural-like units.

- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5(4), 115–133.

Perceptrons and Early Models: Rosenblatt's perceptron (1958) introduced the idea of a trainable linear classifier. Although limited, it marked the first significant step toward learning from data.

- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386–408.

2. Mathematical Challenges and Stagnation

Limitations Identified: Minsky and Papert (1969) detailed the theoretical limitations of single-layer perceptrons. Their critiques led to reduced interest and funding in neural network research.

- Minsky, M., & Papert, S. (1969). Perceptrons: An introduction to computational geometry. MIT Press.

3. Backpropagation and Renewed Interest

Multi-Layer Perceptrons (MLPs): The backpropagation algorithm, popularized by Rumelhart, Hinton, and Williams (1986), offered a practical way to train multi-layer networks and spurred a resurgence of research.

- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533–536.

4. Regularization, Optimization, and Theoretical Insights

Methods to Improve Training: Techniques like dropout (Srivastava et al., 2014), batch normalization (Ioffe & Szegedy, 2015), and advanced optimizers (Kingma & Ba, 2015) helped stabilize and improve the training of deep networks.

- Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929–1958.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML'15) (pp. 448–456).
- Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR'15).

Convolutional Neural Networks (CNNs)

Fukushima's Neocognitron (1980) introduced the concept of local receptive fields and hierarchical feature extraction, setting the stage for modern CNNs. LeCun et al. (1989) demonstrated the effectiveness of applying backpropagation to CNNs, particularly in the domain of handwritten character recognition. Subsequent highly cited work drove the field forward:

- Krizhevsky, Sutskever, and Hinton's AlexNet (2012) was the first CNN to excel on ImageNet, triggering the deep learning revolution in computer vision.
- Simonyan and Zisserman (2015) introduced VGGNet, a very deep CNN model that showed the importance of depth in achieving superior performance.
- He et al. (2016) presented ResNet, an architecture leveraging residual connections to train extremely deep networks effectively, drastically improving accuracy on large-scale image classification tasks.

Papers:

- Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern. 1980;36(4):193-202.
- LeCun Y, Boser B, Denker JS, et al. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1989;1(4):541-551.
- Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, eds. Advances in Neural Information Processing Systems 25. Curran Associates, Inc; 2012:1097-1105.
- Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (ICLR); 2015.
- He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016:770-778.

Recurrent Neural Networks (RNNs)

Elman's (1990) simple RNN was a key innovation for processing sequential data, but these early networks struggled with long-term dependencies due to the vanishing gradient problem. Hochreiter and Schmidhuber's (1997) LSTM architecture overcame this difficulty, allowing RNNs to model longer sequences effectively. Building on LSTMs, Bidirectional RNNs (BiRNNs) and later Bidirectional LSTMs (BiLSTMs) were developed to process input in both forward and backward directions, offering richer contextual understanding in tasks such as speech recognition, natural language processing, and time-series analysis. Cho et al. (2014) introduced the Gated Recurrent Unit (GRU), simplifying the gating mechanisms of LSTMs while retaining their ability to handle long-range dependencies.

Papers:

- Elman JL. Finding structure in time. Cogn Sci. 1990;14(2):179-211.
- Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735-1780.
- Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. 2005;18(5-6):602-610.
- Cho K, van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014:1724-1734.

Edge Artificial Intelligence

Edge Computing & Edge Intelligence

These foundational works introduce the concepts, architectures, and motivations behind moving computation from centralized cloud platforms to distributed edge devices. They also establish how artificial intelligence can be integrated into edge systems to support real-time, resource-constrained civil-engineering applications.

- Shi et al. (2016) present one of the earliest comprehensive discussions of edge computing. The paper defines core principles, motivates the shift toward edge-based processing, and outlines architectures and application domains, including smart cities and transportation systems:
  - Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE internet of things journal*, *3*(5), 637-646
- Deng et al. (2020) introduce the concept of *edge intelligence*, distinguishing between "AI for edge" (optimizing edge infrastructures) and "AI on edge" (running learning and inference directly on edge devices). The paper provides architectures and case studies relevant to structural health monitoring (SHM), transportation networks, and infrastructure systems:
  - Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., & Zomaya, A. Y. (2020). Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet of Things Journal*, *7*(8), 7457-7469.

Edge AI / Edge Computing for Structural Health Monitoring (SHM) & Bridges

These papers focus specifically on SHM and bridge monitoring scenarios where edge computation reduces bandwidth, enables real-time analytics, and supports deployment at scale across civil infrastructure assets.

- Tokognon et al. (2017) survey IoT-based SHM frameworks for bridges, buildings, and other civil structures. Their study highlights network design and processing architectures that naturally motivate edge and fog deployments for large-scale infrastructure monitoring:
    - Tokognon, C. A., Gao, B., Tian, G. Y., & Yan, Y. (2017). Structural health monitoring framework based on Internet of Things: A survey. *IEEE Internet of Things Journal*, *4*(3), 619-635.
- Wang et al. (2023) develop a large-scale wireless IoT monitoring system for the Hong Kong–Zhuhai–Macao Bridge. Edge processors perform local anomaly detection, reducing communication load while supporting near real-time SHM:
    - Wang, X., Wu, W., Du, Y., Cao, J., Chen, Q., & Xia, Y. (2023). Wireless IoT monitoring system in Hong Kong–Zhuhai–Macao bridge and edge computing for anomaly detection. *IEEE Internet of Things Journal*, *11*(3), 4763-4774.
- Bisio et al. (2025) propose a deep-learning–based signal compression framework using convolutional autoencoders deployed directly on edge devices. This approach preserves essential vibration information for damage detection while significantly reducing transmission demands:
    - Bisio, I., Garibotto, C., Lavagetto, F., Sciarrone, A., & Zerbino, M. (2025). Deep at the Edge: AI-Driven Signal Compression for Structural Health Monitoring in Symbiotic IoT Systems. *IEEE Internet of Things Journal*.
- Bisio et al. (2024) apply compressive sensing on low-cost IoT nodes installed on bridges, enabling sparse sampling and partial edge-side processing. The method substantially reduces data volume and energy consumption while preserving structural modal characteristics:
    - Bisio, I., Garibotto, C., Grattarola, A., Lavagetto, F., Sciarrone, A., & Zerbino, M. (2024). Shm with low-cost, low-energy, and low-rate iot devices: Reducing transmission burden with compressive sensing. *IEEE Internet of Things Journal*, *11*(13), 24323-24333.

Edge Intelligence for Transportation Systems (Traffic, Smart Cities)

These works extend edge AI concepts to transportation and smart city applications, emphasizing real-time perception, decision-making, and urban mobility management.

- Gong et al. (2023) provide a comprehensive survey of edge intelligence within intelligent transportation systems (ITS). They discuss architectures, federated and distributed learning paradigms, V2X applications, and real-time operational constraints—all directly relevant to modern transportation engineering:
    - Gong, T., Zhu, L., Yu, F. R., & Tang, T. (2023). Edge intelligence in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *24*(9), 8919-8944.
- Liu et al. (2021) design a smart traffic monitoring system that uses edge servers to run real-time computer vision analytics. The system performs lane-level volume detection and incident identification with reduced latency and network overhead, illustrating how edge AI supports next-generation traffic operations:
    - Liu, G., Shi, H., Kiani, A., Khreishah, A., Lee, J., Ansari, N., ... & Yousef, M. M. (2021). Smart traffic monitoring system using computer vision and edge computing. *IEEE Transactions on Intelligent Transportation Systems*, *23*(8), 12027-12038.

Advanced AI Technologies

Vaswani et al. (2017) introduced the Transformer, a watershed moment in deep learning. By relying on attention mechanisms and discarding recurrent and convolutional operations, Transformers opened the door for models that excelled in natural language processing (NLP) tasks with unprecedented scalability and efficiency. Subsequent highly cited models leveraged this architecture for pre-training and fine-tuning:

- Devlin et al. (2019) released BERT, establishing new benchmarks in a range of NLP tasks through bidirectional pre-training of Transformers.
- Radford et al. (2018) and Brown et al. (2020) pushed the frontier further with GPT-based models and GPT-3, demonstrating remarkable capabilities in few-shot learning and broad language understanding.
- More recently, Touvron et al. (2023) introduced LLaMA, an open-source large language model that aimed to provide researchers and practitioners with more transparent and efficient foundation models, facilitating further advancement in the field.

Papers:

- Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Guyon I, Luxburg UV, Bengio S, et al, eds. Advances in Neural Information Processing Systems 30. Curran Associates, Inc; 2017:6000-6010.
- Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL); 2019:4171-4186.
- Radford A, Narasimhan K, Salimans T, Sutskever I. Improving language understanding by generative pre-training. OpenAI; 2018.
- Brown T, Mann B, Ryder N, et al. Language models are few-shot learners. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H, eds. Advances in Neural Information Processing Systems 33. Curran Associates, Inc; 2020:1877-1901.
- Touvron H, Lavril T, Izacard G, et al. LLaMA: Open and efficient foundation language models. arXiv. Preprint posted online February 22, 2023. arXiv:2302.13971.