CYB-4203/6203

Secure and Trustworthy AI

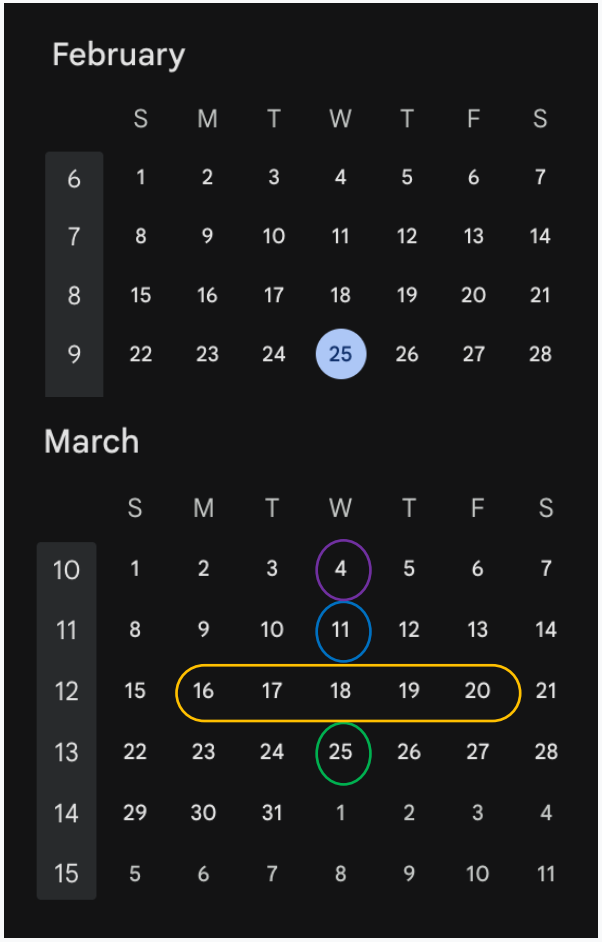
Presentation 9: Bio/Neuro/Psych connections to AI/ML Systems, Lifecycles, and Security

Wednesday, February 25, 2026

Topics: 5.1, 5.2, 5.3

Housekeeping – Schedule Update!

- Assignment 5: Posted Today
- Assignment 6: Posted Mar 4
- I'll be gone to Spain next week for ICISSP conference
  - Monday Mar 2: Dr. Pei Weiping
  - Wednesday Mar 4: Dr. Yi Ting Chua
- Midterm Exam Date: Wednesday March 11
  - Comprehensive through Mar 4
  - I'll post a study guide
- Spring Break: March 16-20
- Midterm Projects Due: Friday March 25 11:59 pm
  - Will incorporate Assignments 5 and 6



Housekeeping – Course Website

https://dallaselleman.github.io/cyb-4203-6203-spring-2026

Today's Agenda

- Material inspired by interactions from our previous session
- 5.1 Biology, neuroscience, and psychology connections with AI/ML systems
- 5.2 Core architecture of modern AI/ML systems and how they differ from traditional software
- 5.3 AI/ML system lifecycle / pipeline: data collection and preparation, model training and evaluation, deployment and integration, monitoring and maintenance
- Assignment 5

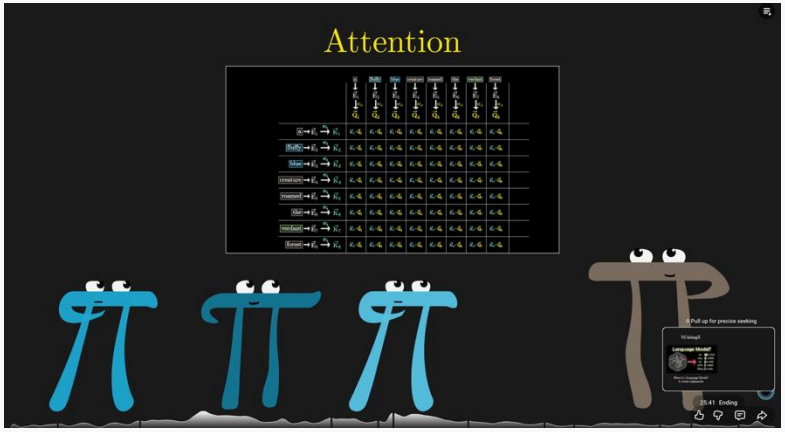
5.1 Bio/neuro/psych connections to AI/ML Systems

Last session: Grant Sanderson (3 Blue 1 Brown)

[Transformers, the tech behind LLMs](#)
Neural Networks Playlist Ch. 5

- Text is broken into tokens, each mapped top an embedding vector in high-dimensional space
- Directions in embedding space encode semantic meaning (woman – man ≈ queen - king)
- Vectors flow through alternating attention blocks and MLP blocks, enriching with context at each step
- All learned behavior lives in weight parameters – 175B in GPT-3
- Final output: a probability distribution over possible next tokens, shaped by temperature

Last session: Grant Sanderson (3 Blue 1 Brown)



[Attention in transformers, step-by-step](#)
Neural Networks Playlist Ch. 6

- Query-Key-Value: each token produces a query ("what I want"), a key ("how to find it"), and a value ("what I get")
- Dot products between queries and keys score relevance; softmax normalizes into weights
- Multi-headed attention (96 parallel heads in GPT-3) captures many types of contextual relationships
- This resolves ambiguity: "mole" means different things depending on surrounding tokens (context)
- Stacking layers builds increasingly abstract representations – from syntax to semantics to reasoning

Last session: “Are LLMs just glorified auto-complete?”

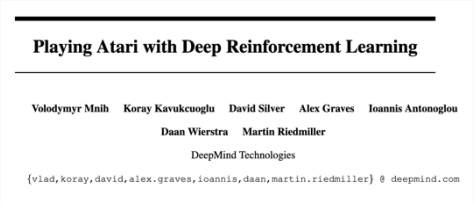
- At the mechanical level, yes: LLMs predict the next token
- But the same reductive logic applies everywhere
    - Neurons are just “electrochemical switches”
    - Computers are just “flipping bits”
- The important thing is the **emergence of novel behaviors at scale**
- GPT-3: 175B parameters.
- Human brain: 86B neurons, ~100T synaptic connections
- Emergence: complex capabilities arising from the composition of simple operations
- Example: BOIDS – 3 simple rules (Cohesion, Separation, Alignment) combine to produce flocking behavior
- Original author: Craig Reynolds (amazing webpage)

Boids - The Emergence of Flocks

Coding Adventure: Boids

Last session: AI reflection i.e., windshield time / shower thoughts

- Related neuroscience / psychology: Default Mode Network, Incubation
- During sleep, the brain replays experiences and consolidates memories (hippocampal replay)
- DeepMind's DQN (2015): Experience replay – re-sampling past experiences during training – directly inspired by neuroscience
- Deep Reinforcement Learning: Combines deep neural networks (DNN) with reinforcement learning (RL)
- Offline consolidation: models reorganize internal representations between training phases, analogous to memory consolidation during sleep
- Complementary learning systems: fast learning (hippocampus) + slow integration (neocortex) inspires dual-system AI architectures



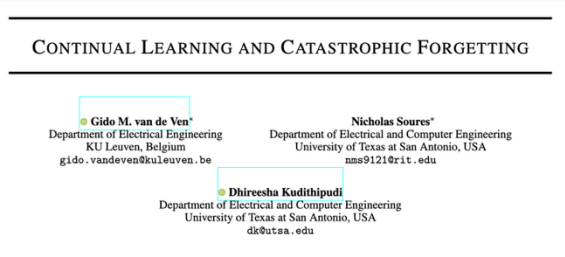
Original DQN paper

Google DeepMind blog

'The Power of Self-Learning Systems' - Demis Hassabis lecture at IAS

Last Session: Continual Learning and Catastrophic Forgetting

- Humans learn continuously; learning to read doesn't make you forget how to walk
- Neural networks suffer catastrophic forgetting – new training can overwrite previous knowledge
- Biological solution: synaptic consolidation, complementary learning systems, sleep-based replay
- AI approaches inspired by biology: elastic weight consolidation, progressive networks, rehearsal, etc.
- An active research frontier – and directly relevant to AI security (retraining risks, model drift)



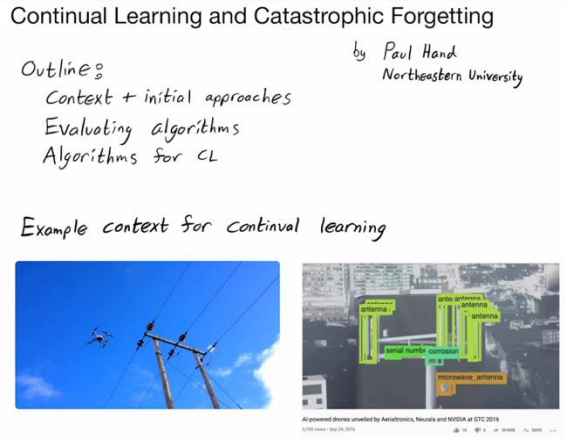
CONTINUAL LEARNING AND CATASTROPHIC FORGETTING

Gido M. van de Ven*
Department of Electrical Engineering
KU Leuven, Belgium
gido.vandeven@kuleuven.be

Nicholas Soures*
Department of Electrical and Computer Engineering
University of Texas at San Antonio, USA
nms9121@rit.edu

Dhireesha Kudithipudi
Department of Electrical and Computer Engineering
University of Texas at San Antonio, USA
dk@utsa.edu

[Arxiv paper link]



Continual Learning and Catastrophic Forgetting

by Paul Hand
Northeastern University

Outline:
Context + initial approaches
Evaluating algorithms
Algorithms for CL

Example context for continual learning

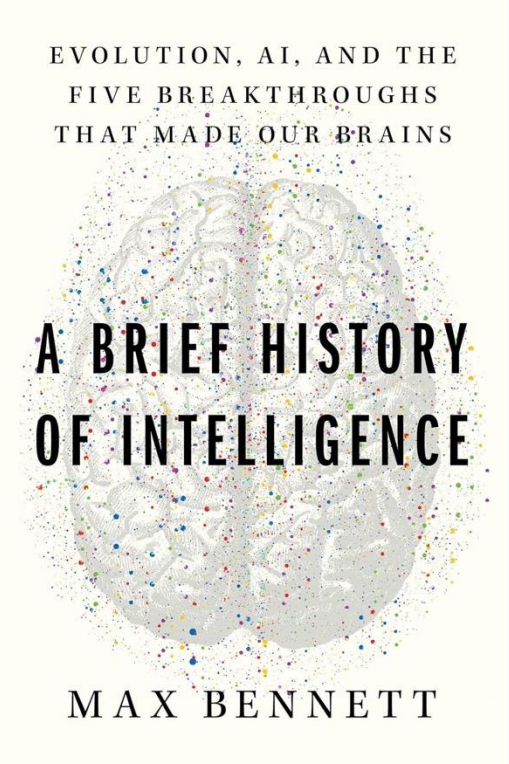
[Paul Hand (Northeastern University) video]

Last Session: Biological / Artificial Intelligence Feedback Loop

- AI borrows not just *metaphors,* but mechanisms from biology
- The influence between neuroscience and AI runs both directions; advances in one unlock new questions in the other
- **Biology → AI examples:** Neurons → Perceptron; Visual Cortex → CNNs (cat video, Nobel Lecture); Hippocampal Replay → Experience Replay; Cognitive Attention → Transformer Attention
- **AI → Biology examples:** DNNs now predict neural responses better than hand-built models; Transformer attention generates new hypotheses about how brains allocate processing
- Current frontier: 'in silico' neuroscience – using AI models to understand biological brains

AMAZING Book: A Brief History of Intelligence – Max Bennett

- Discusses 5 intelligence breakthroughs in biological evolution, makes connections to developments and analogs in AI
    - Steering (chemical gradient valence stimulus in first bilaterians, ~550M years ago)
    - Reinforcement learning (dopamine-driven loops in basal ganglia in first vertebrates ~500M years ago)
    - Simulation (generative models in neocortices of first mammals, ~200M years ago)
    - Mentalizing (social intelligence and theory of mind in first primates ~50-60M years ago)
    - Symbolic language (communication and grammar in early humans, ~100-300K years ago)
- Online slow-flip version - Spotify audiobook
  Medium article: Book review



Why Biological / Artificial Intelligence Matters for CYB-4203-6203

- Understanding what AI is and isn't ("just autocomplete") is prerequisite to reasoning about its security

- Biology, neuroscience, and psychology can inform and illuminate AI capabilities, vulnerabilities, and failure modes (and vice versa)

- Examples:
  - Continual learning challenges create model drift and retraining vulnerabilities
  - AI's black box problem ↔ neurosci's challenge understanding biological brains
  - Prompt injection ↔ deception and lying
  - Data poisoning ↔ human misinformation & belief formation

5.2 AI/ML system architectures: Contrast with traditional software

Traditional Software vs. AI/ML: Design Philosophy

Traditional Software	AI/ML
Encoded Decisions	Approximated Functions
Behavior defined by explicit rules & logic	Behavior learned from data and training
“If X, then Y” – every decision path is authored by a human	“Given enough examples of X to Y, learn the mapping.”
Deterministic: same input always produces same output (mostly)	Probabilistic: outputs are predictions with confidence levels
Correctness is provable for many domains	Correctness is statistical, not absolute

Traditional Software vs. AI/ML: Development Process

Traditional Software	AI/ML
Requirements, design, code, test	Data collection, training, evaluation, tuning
Software is authored line-by-line	Models are trained, not written (i.e., grown not built)
Version control tracks every change	Experiment tracking (hyperparameters, datasets, metrics)
Code review ensures quality	Evaluation on held-out test sets
Developer skill determines capability	Data quality and quantity determine capability

Traditional Software vs. AI/ML: Failure Modes & Debugging

Traditional Software	AI/ML
Bugs are reproducible	Failures are often probabilistic
Stack traces point to the failing ‘line of code’	No ‘line of code’ to blame
Logic errors can be stepped through	Model behavior emerges from training data and architecture
Root cause is identifiable	Interpretability is an active research challenge
Fix the code to fix the bug	Fix the data? The architecture? The training process?

Traditional Software vs. AI/ML: Testing & Verification

Traditional Software	AI/ML
Unit tests, integration tests, end-to-end tests	Evaluation metrics: accuracy, precision, recall, F1
Formal verification for critical systems	Benchmark datasets and leaderboards
Code coverage as a quality metric	Adversarial testing and red-teaming
You can prove a sorting algorithm correct	You cannot prove a classifier will never misclassify

Traditional Software vs. AI/ML: Advantages & Use Cases

Traditional Software	AI/ML
Well-defined business rules	Pattern recognition in unstructured data
Regulatory compliance logic	Natural language understanding and generation
Deterministic calculations	Computer vision and image analysis
Transaction processing	Anomaly detection in complex systems
Real-time control systems	Tasks too complex to hand-code rules for

5.3 – AI/ML system Lifecycle / Pipeline

AI/ML System Lifecycle / Pipeline

Data Collection & Preparation	Model Training & Evaluation	Deployment & Integration	Monitoring & Maintenance
Data defines what the model learns – and what biases it inherits	Architecture choice determines what patterns the model can learn	Inference: running a trained model on new inputs to produce predictions	Data drift: the real world changes, but the model was trained on yesterday's data
Feature engineering transforms raw data into model-ready representations	Training requires massive compute: GPUs/TPUs running for days to months	Latency vs. accuracy tradeoffs: model compression, quantization, distillation	Concept drift (input – output relationships), continual learning, catastrophic forgetting
Data versioning: tracking which data produced which model	Model weights are the learned parameters – often the most valuable intellectual property	Deployment targets: cloud APIs, on-device (mobile, IoT), embedded systems	Track accuracy, latency, fairness, input distribution
Security implications: data poisoning, privacy leakage, supply chain risks	Security implications: model theft, embedded backdoor attacks	Security implications: adversarial inputs, prompt injection, model inversion, resource exhaustion	Security implications: feedback manipulation, blind spot monitoring

Assignment 5: Published later today