Module 20: Advanced Gen AI and LLMs

Learning outcomes:

1. Evaluate strategies for scaling large language models, including hyperparameter tuning, performance trade-offs, cost and robustness.
2. Analyse emergence thresholds and evaluate their practical and ethical risks for task-specific performance.
3. Assess the real-world value of few-shot learning in accelerating training and deployment across industries.
4. Analyse the foundational components of transformers and compare building a model from scratch with using pre-built models.
5. Refine transformer architectures in PyTorch.

Scaling strategies and efficient training

- Large transformers require advanced optimisation (Adam/AdamW, schedules) and deliberate hardware planning. Stability and throughput depend on batch/sequence choices and parallelism.
- Scaling laws (empirical, power law trends) guide cost-effective scaling across parameters, data and compute. Gains are sublinear (each doubling tends to deliver smaller absolute improvements).
- Best results come from balanced scaling (not just ‘bigger model’). For a fixed budget, shifting some capacity into more/better data can outperform parameter-only growth.
- Hyperparameters matter more at scale (learning rate, batch size, regularisation). They affect robustness and reliability, not just headline accuracy.

Emergence

- Emergent behaviours are capabilities that become reliably observable only after crossing size/data thresholds. They are not guaranteed by small, incremental changes.
- Planning must allow for discontinuous jumps in capability. Evaluate continuously and task-specifically rather than assuming smooth scaling.
- Practical/ethical implications: emergence increases unpredictability, raises resource and environmental costs and intensifies safety, fairness and governance challenges. Organisations need stronger oversight and evaluation.

Transformer essentials

The core components of transformers are embeddings, positional encoding, self-attention/multi-head attention and feed-forward blocks. Training stability depends on residual connections and normalisation.

Why and when to build vs use pre-trained

- Understanding transformer internals improves diagnosis, fine-tuning and responsible deployment – valuable even if you never train from scratch.
- Choosing between pre-trained and from-scratch hinges on task specificity, data availability, compute budget and time-to-value. Pre-trained is often the practical default, with from-scratch reserved for specialised needs.