NEURAL NETWORKS FROM SCRATCH
LECTURE 04 - PROJECT PROPOSALS

Hendrik Borras, Robin Janssen
{hendrik.borras, robin.janssen}@ziti.uni-heidelberg.de,
HAWAII Lab, Institute of Computer Engineering
Heidelberg University

RECAP: ANNS & GPUS

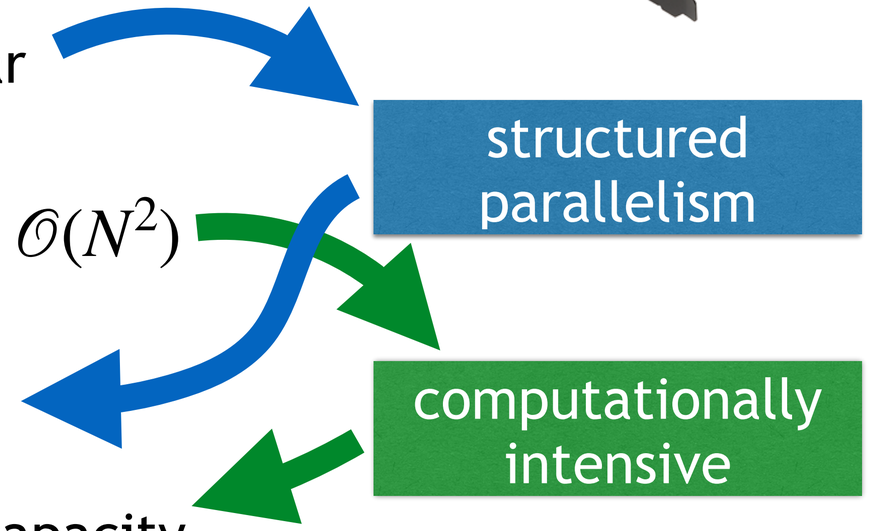
NVIDIA

Matrix multiplication $Y = W \cdot X$ is at the heart of neural networks

- N^2 elements - independent but very similar computations
- Compute scales with $\mathcal{O}(N^3)$, memory with $\mathcal{O}(N^2)$

GPUs are

- massively parallel vector-based processors
- excellent in compute, limited in memory capacity
- Bonus: highly energy efficient => many operations per Watt



PROJECT WORK

FORMALITIES

Total “Arbeitsaufwand” for the practical: 180h (FP: 240h)

- About 20-30h are covered by lecture and exercises
- Expect to spend another 30h on poster preparation and final report
- → 120h remaining for the project work (FP: 180h)

Group Projects

- Proposed by the groups in a project proposal
- General plan to be discussed with us before starting
- Projects should be built such that one can distinguish individual work
- Consulting with us possible in the usual time slot (not necessary, but recommended)
- Coding with LLMs is fine, but focus on learning & understanding, not flashy results

PROJECT PROPOSALS

Due **Tuesday, 18.11.2025, 9:00 AM** (**proposal & exercise**)

Topic Choice

- Feel free to choose a topic that interests you, or choose from the examples
- Only condition: Should be in the “from scratch” spirit

Content

- Title & Abstract
- Relevant literature
- Work packages with description, estimated duration, deliverables
- No longer than 1-2 pages, should show that you put some thought into the project
- ChatGPT et al. are very good at project planning, but please verify that the proposed steps, content and workload are realistic

EXAMPLE PROJECTS

EXAMPLE 1: EXPLORING OPTIMIZERS

Goal

Implement modern optimizers to understand their behavior and trade-offs

Key tasks

Implement several optimizers (Adam, RAdam, RMSProp, Shampoo, Muon, etc.)

Compare convergence speed and final accuracy on the same model/dataset, e.g., an MLP on CIFAR10

Extension: Schedulers, second order optimisers

Why it is cool

ANNs (or other things) learning fast & smart

---

input : γ (lr), β_1, β_2 (betas), θ_0 (params), $f(\theta)$ (objective)
λ (weight decay), *amsgrad*, *maximize*
initialize : $m_0 \leftarrow 0$ (first moment), $v_0 \leftarrow 0$ (second moment), $\widehat{v_0}^{max} \leftarrow 0$

---

for $t = 1$ **to** $\ldots$ **do**
  if *maximize* :
    $g_t \leftarrow -\nabla_\theta f_t(\theta_{t-1})$
  else
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$
  if $\lambda \neq 0$
    $g_t \leftarrow g_t + \lambda \theta_{t-1}$
  $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$
  $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
  $\widehat{m_t} \leftarrow m_t / \big(1 - \beta_1^t\big)$
  $\widehat{v_t} \leftarrow v_t / \big(1 - \beta_2^t\big)$
  if *amsgrad*
    $\widehat{v_t}^{max} \leftarrow \max(\widehat{v_t}^{max}, \widehat{v_t})$
    $\theta_t \leftarrow \theta_{t-1} - \gamma \widehat{m_t} / \big(\sqrt{\widehat{v_t}^{max}} + \epsilon\big)$
  else
    $\theta_t \leftarrow \theta_{t-1} - \gamma \widehat{m_t} / \big(\sqrt{\widehat{v_t}} + \epsilon\big)$

---

return θ_t

EXAMPLE 1: WORK PACKAGES

WP	Description	Content	Estimated Duration (hrs)
WP1	Literature Review and Baseline Setup	Survey modern optimizers, summarise update rules, implement baseline training	40
WP2	Implementation of Optimizers	Implement chosen optimizers manually	50
WP3	Experimental Evaluation Framework	Set up systematic benchmarking pipeline: fixed datasets, architectures, metrics	40
WP4	Comparative Study and Analysis	Perform quantitative and qualitative comparison of all implemented optimizers	40
WP5	Extension / Exploration	Optional - Implement one additional, more advanced idea (e.g.: LR scheduler)	30

Total: 200 hrs -> slightly ambitious for 2, a bit low for 3

EXAMPLE 2: TOOLS OF THE TRADE

Goal

Build and investigate training refinement strategies

Key Tasks

Implement Learning Rate Schedulers

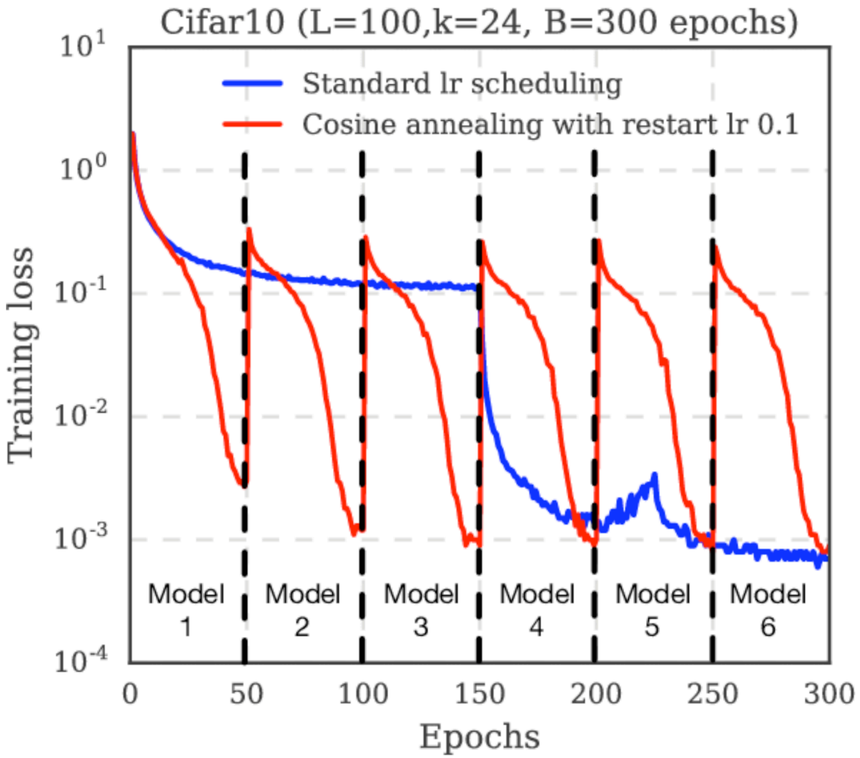
Add Regularization Techniques

Implement Data Augmentation Pipeline

Extension: Improved Optimizer, combination effects

Why it is cool

They may seem small, but these “tricks” add up to deliver the performance of today’s ANNs



EXAMPLE 3: CNNS FROM SCRATCH

Goal

Implement and train a CNN end-to-end

Key tasks

Implement convolutional and pooling layers manually

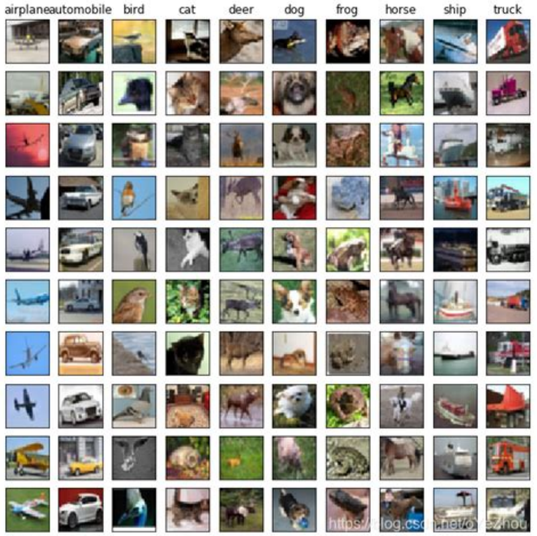
Build up to a small CNN architecture (LeNet, MiniVGG, UNet,…)

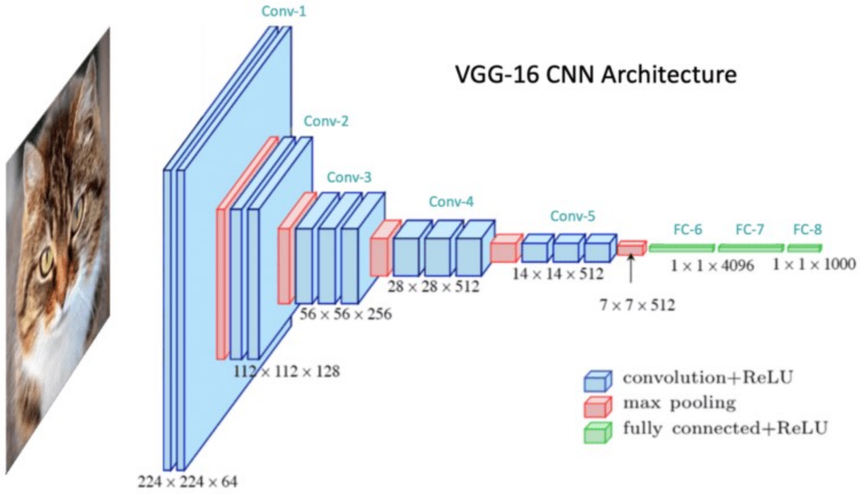
Train and evaluate on CIFAR-10 or Tiny ImageNet.

Extensions: Advanced architectures, advanced applications

Why it is cool

Foundational for a whole field of applications, potentially very nice visualisations

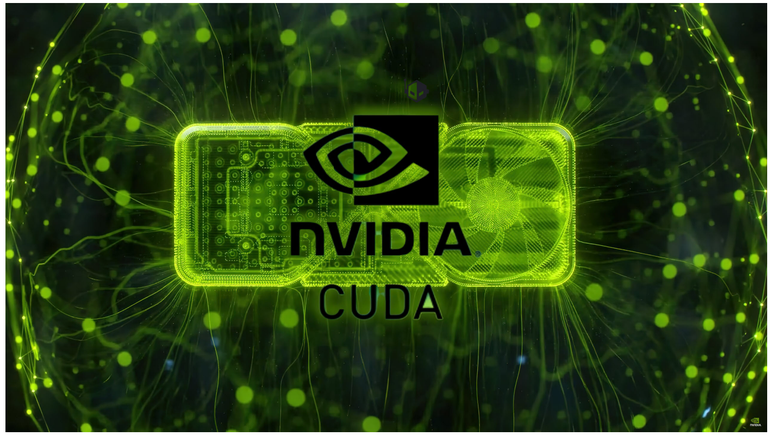




EXAMPLE 4: LOW-LEVEL DEEP LEARNING

Goal

Implement key DNN operations directly in a low-level language



Key tasks

Implement custom CUDA (or Triton) kernels for linear layers, activation functions

Explore kernel fusion to reduce launch and synchronization overhead

Profile GPU performance and compare to CuPy/PyTorch

Extensions: Custom ops, study memory vs. compute bottlenecks

Why it is cool

Very much in the “from scratch” spirit

EXAMPLE 5: TRANSFORMERS FROM SCRATCH

Goal

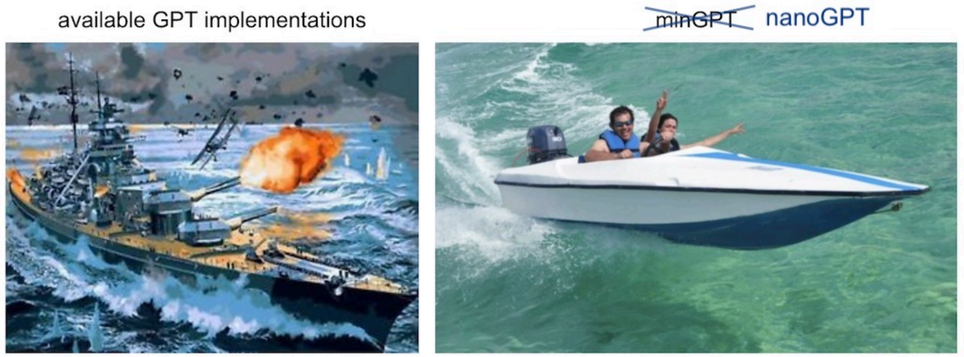
Implement the core components of a transformer, build and train NanoGPT

Key tasks

Implement multi-head self-attention, layer normalization, embeddings

Derive and code backward passes for attention and feedforward layers

Train a small language model



Why it is cool

Have you heard about these things called LLMs?

Note

While potentially very rewarding, this project is a lot of work

Recommended for FP, AP only if you are willing to put in a lot of effort as a group

We can offer code templates and tests for forward/backward prop

EXAMPLE 6: MODEL COMPRESSION STUDY

Goal

Explore model compression through reduced precision and structured pruning

Key tasks

Implement fixed-point weight quantisation (e.g., 8-bit, 4-bit)

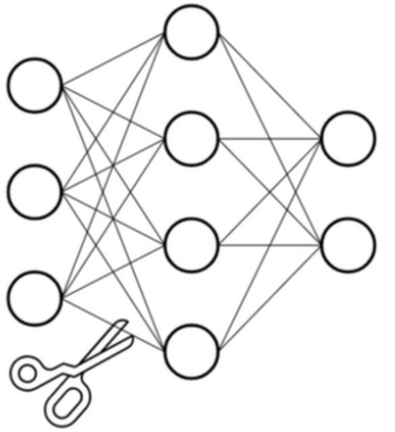
Prune weights or neurons based on magnitude or gradient criteria

Implement post-training quantisation and quantisation-aware training

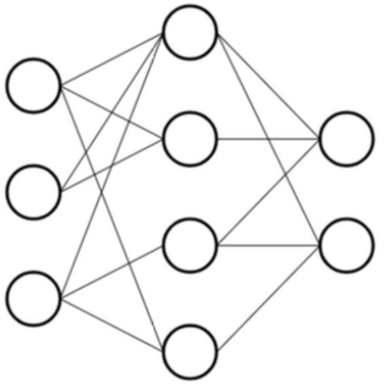
Extensions: Visualisations, quantifying efficiency vs. Accuracy

Why it is cool

Efficiency is crucial, investigate core building blocks of ANNs closely



Before pruning

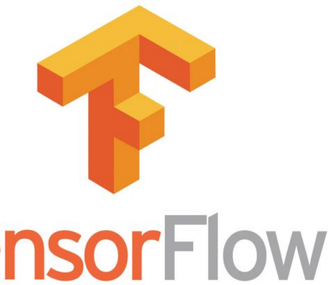


After pruning

EXAMPLE 7: BUILDING AN AUTOGRAD ENGINE

Goal

Design a minimal automatic differentiation system for ANN training



Key tasks

Implement a computational graph with forward and backward passes

Define gradient propagation through primitive operations (add, mul, etc.)

Build a simple MLP on top of your autograd system

Extensions: Visualisation, gradient checking

Why it is cool

This is the foundation of ML—if everyone had to write the backward pass manually all the time, we would not have ChatGPT today

EXAMPLE 8: MANIPULATING ANNS

Goal

Use gradients to interpret and manipulate neural networks.

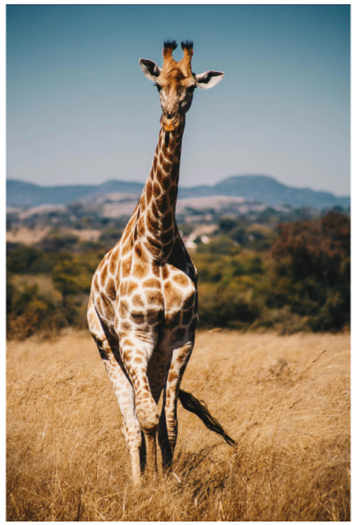
Key tasks

Compute gradients of loss w.r.t. input images (adversarial examples)

Implement FGSM or PGD adversarial attacks.

Implement GradCAM for visualizing decision regions

Extensions: Investigate robustness through adversarial training



Why it is cool

Interpretability is crucial in many applications, adversarial attacks pose security risks with growing importance of ANNs

WRAPPING UP

SUMMARY

Group project formalities

- 120 (180 FP) hours per person
- From scratch spirit
- Consulting possible in usual time slot - not necessary, but recommended

Project proposal

- **Due: Tuesday, 18.11.2025, 9:00 AM (together with ex03)**
- Title, abstract, literature, work packages, rough time plan
- Short, but show proper planning

ANY QUESTIONS?