Efficient AI / Efficient ML -- Graduate Course Project Paper List

> **Course:** CAP6614 -- Efficient AI **Compiled:** February 2026 **Scope:** ~50 influential papers (2023--2025) from top venues **Selection criteria:** High citation count, public code availability, reproducibility within a semester project, focus on large-model efficiency (LLMs, diffusion models, VLMs)

Category 1: LLM Pruning and Knowledge Distillation (8 papers)

1.1 SparseGPT: One-Shot LLM Pruning

- **Title:** SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot
- **Authors:** Frantar, E. and Alistarh, D.
- **Venue:** ICML 2023
- **Description:** First method to prune LLMs (100B+ parameters) to 50-60% sparsity in one shot without any retraining, using an efficient approximate sparse regression solver. Achieves negligible perplexity increase at 50% unstructured sparsity on OPT-175B and BLOOM-176B.
- **Project angle:** Apply SparseGPT to prune LLaMA-2-7B or Mistral-7B at various sparsity levels, benchmark perplexity and zero-shot accuracy, and combine with quantization (sparse + quantized). Code available at github.com/IST-DASLab/sparsegpt.

1.2 Wanda: Simple and Effective LLM Pruning

- **Title:** A Simple and Effective Pruning Approach for Large Language Models
- **Authors:** Sun, M., Liu, Z., Bair, A., and Kolter, J. Z.
- **Venue:** ICLR 2024
- **Description:** Proposes Wanda (Weights AND Activations), an extremely simple pruning method that selects weights to prune based on the product of weight magnitude and input activation norm. Matches or exceeds SparseGPT quality with orders of magnitude less computation (no weight updates needed).
- **Project angle:** Implement Wanda's pruning criterion (just a few lines of code), compare against magnitude pruning and SparseGPT on LLaMA-2-7B, and analyze which layers are most sensitive to pruning. Code available at github.com/locuslab/wanda.

1.3 LLM-Pruner: Structural Pruning for LLMs

- **Title:** LLM-Pruner: On the Structural Pruning of Large Language Models
- **Authors:** Ma, X., Fang, G., and Wang, X.
- **Venue:** NeurIPS 2023
- **Description:** Proposes a task-agnostic structural pruning framework for LLMs that selectively removes non-critical coupled structures based on gradient information. Preserves the multi-task solving and language generation ability of the original LLM with minimal retraining (3 hours with LoRA).

- **Project angle:** Apply LLM-Pruner to prune LLaMA-2-7B or LLaMA-3-8B at different compression ratios (20%, 30%, 50%), compare with unstructured pruning (SparseGPT, Wanda), and measure actual inference speedup (not just FLOPs). Code available at github.com/horseee/LLM-Pruner.

1.4 SliceGPT: Compression by Deleting Rows and Columns

- **Title:** SliceGPT: Compress Large Language Models by Deleting Rows and Columns
- **Authors:** Ashkboos, S., Croci, M. L., do Nascimento, M. G., Hoefler, T., and Hensman, J.
- **Venue:** ICLR 2024
- **Description:** Compresses LLMs by applying orthogonal projections to each layer's weight matrices, then deleting rows and columns corresponding to the smallest-magnitude components. Removes up to 25% of rows/columns in LLaMA-2-70B with minimal perplexity degradation and real wall-clock speedup.
- **Project angle:** Apply SliceGPT to LLaMA-2-7B or Phi-2 at different slicing levels, compare with other structural pruning methods, and benchmark the actual end-to-end speedup on GPU inference.

1.5 Sheared LLaMA: Pruning to Accelerate Pretraining

- **Title:** Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning
- **Authors:** Xia, M., Gao, T., Zeng, Z., and Chen, D.
- **Venue:** ICLR 2024
- **Description:** Develops targeted structured pruning that prunes a larger LLM to a specified target shape by removing layers, heads, and intermediate/hidden dimensions, combined with dynamic batch loading that adjusts data composition based on domain losses. Produces strong 1.3B and 2.7B models from LLaMA-2-7B with only 3% of the original pretraining compute.
- **Project angle:** Apply the Sheared LLaMA approach to prune a 7B model to different target sizes (1B, 2B, 3B), compare with training from scratch at the same size, and evaluate on downstream benchmarks. Code available at github.com/princeton-nlp/LLM-Shearing.

1.6 Minitron: LLM Pruning and Distillation in Practice

- **Title:** Compact Language Models via Pruning and Knowledge Distillation
- **Authors:** Muralidharan, S., Sreenivas, S., et al. (NVIDIA)
- **Venue:** NeurIPS 2024
- **Description:** Develops practical compression best practices combining depth pruning, width pruning (attention heads, MLP dimensions), and knowledge distillation-based retraining. Derives 8B and 4B models from Nemotron-4 15B requiring up to 40x fewer training tokens than training from scratch. The resulting Minitron models outperform Mistral-7B, Gemma-7B, and match LLaMA-3-8B.
- **Project angle:** Apply the Minitron pruning strategy (depth vs. width pruning comparison) to a smaller open-source model, evaluate the iterative pruning approach, and measure quality-efficiency trade-offs. Code available at github.com/NVlabs/Minitron.

1.7 MiniLLM: Knowledge Distillation for LLMs

- **Title:** MiniLLM: Knowledge Distillation of Large Language Models
- **Authors:** Gu, Y., Dong, L., Wei, F., and Huang, M.
- **Venue:** ICLR 2024

- **Description:** Proposes reverse KL divergence-based knowledge distillation for LLMs, addressing the exposure bias problem in standard KD for autoregressive models. Shows that minimizing reverse KL prevents the student from overestimating low-probability regions of the teacher distribution, achieving better generation quality than forward KL.
- **Project angle:** Distill a LLaMA-2-7B teacher into a smaller student (e.g., TinyLLaMA-1.1B) using the proposed reverse-KL objective, compare against standard KD and direct training, and evaluate on text generation benchmarks.

1.8 ShortGPT: Layer Redundancy in LLMs

- **Title:** ShortGPT: Layers in Large Language Models are More Redundant Than You Expect
- **Authors:** Men, X., Xu, M., Zhang, Q., Wang, B., Lin, H., Lu, Y., Han, X., and Chen, W.
- **Venue:** arXiv 2024 (highly cited)
- **Description:** Introduces Block Influence (BI) as a metric to measure layer importance in LLMs, revealing that many layers are highly redundant. Simply removing the least important 25% of layers in LLaMA-2-13B yields surprisingly small perplexity increases, suggesting depth pruning is underexplored.
- **Project angle:** Compute BI scores for different LLM families (LLaMA, Mistral, Phi), reproduce the layer-removal experiments, and compare with more sophisticated pruning methods to understand when simple depth pruning suffices.

Category 2: Quantization (9 papers)

2.1 GPTQ: Post-Training Quantization for LLMs

- **Title:** GPTQ: Accurate Post-Training Quantization for Generative Pre-Trained Transformers
- **Authors:** Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D.
- **Venue:** ICLR 2023
- **Description:** Proposes a one-shot weight quantization method for LLMs based on approximate second-order information (Optimal Brain Quantization), enabling accurate 3-bit and 4-bit quantization of GPT-scale models (175B parameters) in a few hours on a single GPU, with minimal perplexity degradation.
- **Project angle:** Apply GPTQ to quantize LLaMA-2-7B or LLaMA-3-8B to 4-bit/3-bit, benchmark perplexity on WikiText-2 and C4, measure inference speedup on GPU, and compare with AWQ. Code widely available (AutoGPTQ, github.com/AutoGPTQ/AutoGPTQ).

2.2 AWQ: Activation-Aware Weight Quantization

- **Title:** AWQ: Activation-Aware Weight Quantization for LLM Compression and Acceleration
- **Authors:** Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang, W.-C., Xiao, G., Dang, X., Gan, C., and Han, S.
- **Venue:** MLSys 2024
- **Description:** Observes that only ~1% of weight channels (the "salient" channels determined by activation magnitudes) are critical for LLM performance. Proposes protecting these channels via per-channel scaling before quantization, achieving superior 4-bit quantization quality without backpropagation or reconstruction.

- **Project angle:** Implement the activation-aware scaling strategy for LLaMA-2-7B or LLaMA-3-8B, compare against GPTQ and round-to-nearest baselines, and deploy the 4-bit model with the TinyChat inference engine. Code at github.com/mit-han-lab/llm-awq.

2.3 SqueezeLLM: Dense-and-Sparse Quantization

- **Title:** SqueezeLLM: Dense-and-Sparse Quantization
- **Authors:** Kim, S., Hooper, C., Gholami, A., Dong, Z., Li, X., Shen, S., Mahoney, M. W., and Keutzer, K.
- **Venue:** ICML 2024
- **Description:** Introduces a sensitivity-based non-uniform quantization approach that combines dense low-bit quantization for the majority of weights with a sparse representation for outlier weights. Achieves significant improvements at ultra-low bit-widths (3-bit), outperforming uniform quantization methods.
- **Project angle:** Apply SqueezeLLM to quantize LLaMA models at 3-bit and 4-bit, compare the dense-and-sparse approach against purely dense quantization (GPTQ, AWQ), and analyze the distribution and importance of outlier weights.

2.4 QuIP#: Quantization with Incoherence Processing

- **Title:** QuIP#: Even Better LLM Quantization with Hadamard Incoherence and Lattice Codebooks
- **Authors:** Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and Sa, C. D.
- **Venue:** ICML 2024
- **Description:** Advances extreme LLM quantization by using Hadamard rotations to make weight matrices incoherent (spreading outliers) and lattice codebooks for efficient vector quantization. Achieves state-of-the-art results at 2-bit quantization, where prior methods fail catastrophically.
- **Project angle:** Apply QuIP# to quantize LLaMA-2-7B at 2-bit and 3-bit, compare against GPTQ and AQLM at extreme compression levels, and analyze the role of Hadamard incoherence in spreading weight outliers.

2.5 AQLM: Additive Quantization for Extreme Compression

- **Title:** Extreme Compression of Large Language Models via Additive Quantization
- **Authors:** Egiazarian, V., Panferov, A., Kuznedelev, D., Frantar, E., Babenko, A., and Alistarh, D.
- **Venue:** ICML 2024
- **Description:** Adapts additive quantization (from information retrieval) for LLM compression, representing each weight vector as a sum of entries from multiple learned codebooks. First scheme that is Pareto optimal in accuracy-vs-model-size at less than 3 bits per parameter, dramatically improving the 2-bit quantization frontier.
- **Project angle:** Apply AQLM to compress LLaMA-2-7B to 2-bit, compare with QuIP# and GPTQ at the same bit-width, and measure the trade-off between codebook size, inference speed, and perplexity. Code at github.com/Vahe1994/AQLM.

2.6 OmniQuant: Omnidirectionally Calibrated Quantization

- **Title:** OmniQuant: Omnidirectionally Calibrated Quantization for Large Language Models
- **Authors:** Shao, W., Chen, M., Zhang, Z., Xu, P., Zhao, L., Li, Z., Zhang, K., Xu, P., Liu, H., and Luo, P.
- **Venue:** ICLR 2024

- **Description:** Proposes learnable weight clipping (LWC) and learnable equivalent transformation (LET) that jointly optimize quantization parameters in a differentiable manner, requiring only block-wise optimization and a small calibration set. Supports both weight-only and weight-activation quantization down to W2A16 and W4A4 settings.
- **Project angle:** Apply OmniQuant to LLaMA-2-7B in both weight-only (W4A16, W3A16) and weight-activation (W4A4) modes, compare with GPTQ and AWQ, and evaluate on both perplexity and downstream task accuracy.

2.7 BitNet b1.58: Ternary LLMs

- **Title:** The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits
- **Authors:** Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J., and Wei, F.
- **Venue:** arXiv 2024 (Microsoft Research, extremely high impact)
- **Description:** Introduces BitNet b1.58, a 1.58-bit (ternary: {-1, 0, 1}) LLM architecture that matches full-precision LLaMA performance starting from a 3B parameter scale while offering dramatic memory, latency, and energy savings. Replaces all matrix multiplications with integer additions.
- **Project angle:** Implement ternary weight training for a small transformer language model (125M-350M parameters), measure perplexity vs. the full-precision baseline, and profile theoretical and actual throughput gains.

2.8 Q-Diffusion: Quantizing Diffusion Models

- **Title:** Q-Diffusion: Quantizing Diffusion Models
- **Authors:** Li, X., Liu, Y., Lian, L., Yang, H., Dong, Z., Kang, D., Zhang, S., and Keutzer, K.
- **Venue:** ICCV 2023
- **Description:** First work to systematically study post-training quantization for diffusion models, identifying that the time-step varying activation distributions create unique challenges. Proposes calibration strategies and a split shortcut technique to enable 4-bit weight quantization of diffusion models.
- **Project angle:** Apply PTQ to a pretrained Stable Diffusion model, compare image quality (FID) at 8-bit vs. 4-bit quantization, and analyze how different timesteps respond to quantization.

2.9 QTIP: Trellis Quantization for LLMs

- **Title:** QTIP: Quantization with Trellises and Incoherence Processing
- **Authors:** Tseng, A., Sun, Q., Hou, D., and De Sa, C.
- **Venue:** NeurIPS 2024
- **Description:** Extends QuIP# with trellis-coded quantization, using convolutional codes to define a rich set of codebooks that can be decoded efficiently. Achieves the best known 2-bit LLM quantization results with practical inference speedups, further closing the gap with full-precision models.
- **Project angle:** Compare QTIP with QuIP# and AQLM at 2-bit quantization on LLaMA-2-7B/13B, benchmark inference throughput, and analyze the impact of different trellis structures on quality.

Category 3: Efficient LLM Inference and Serving (10 papers)

3.1 FlashAttention-2: Fast and Memory-Efficient Attention

- **Title:** FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning
- **Authors:** Dao, T.
- **Venue:** ICLR 2024
- **Description:** Improves on FlashAttention by optimizing the parallelism strategy (parallelizing over sequence length instead of batch/heads), reducing non-matmul FLOPs, and better utilizing GPU hardware. Achieves up to 2x speedup over FlashAttention-1 and approaches 50-73% of theoretical peak FLOP utilization on A100 GPUs.
- **Project angle:** Benchmark FlashAttention-2 at various sequence lengths (2K-16K-32K), profile memory usage, measure wall-clock speedup vs. standard attention and FlashAttention-1, and write a technical analysis of the tiling and work-partitioning strategies. Code at github.com/Dao-AILab/flash-attention.

3.2 Speculative Decoding: Lossless LLM Acceleration

- **Title:** Fast Inference from Transformers via Speculative Decoding
- **Authors:** Leviathan, Y., Kalman, M., and Matias, Y.
- **Venue:** ICML 2023
- **Description:** Proposes speculative decoding where a small draft model generates candidate tokens that are verified in parallel by the large target model, providing lossless speedup (the output distribution is mathematically identical to standard decoding). Achieves 2-3x speedup for large models.
- **Project angle:** Implement speculative decoding with different draft/target model pairs (e.g., LLaMA-2-7B drafting for LLaMA-2-70B, or Phi-2 for LLaMA-3-8B), tune the number of speculative tokens, and measure acceptance rates and wall-clock speedup across text domains.

3.3 EAGLE: Feature-Level Speculative Sampling

- **Title:** EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty
- **Authors:** Li, Y., Wei, F., Zhang, C., and Zhang, H.
- **Venue:** ICML 2024
- **Description:** Instead of training a separate draft model, EAGLE trains a lightweight head that auto-regressively predicts the second-top-layer features of the target LLM, addressing the uncertainty problem in next-feature prediction by integrating tokens from one step ahead. Achieves ~3x lossless speedup, significantly outperforming Medusa and standard speculative decoding.
- **Project angle:** Deploy EAGLE on LLaMA-2-7B/13B or Vicuna models, compare speedup against vanilla speculative decoding and Medusa, and analyze how acceptance rates vary across different generation tasks (code, math, open-ended). Code at github.com/SafeAILab/EAGLE.

3.4 Medusa: Multiple Decoding Heads for Parallel Generation

- **Title:** Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads
- **Authors:** Cai, T., Li, Y., Geng, Z., Peng, H., Lee, J. D., Chen, D., and Dao, T.
- **Venue:** ICML 2024
- **Description:** Adds multiple parallel decoding heads to an LLM that predict several future tokens simultaneously, then uses a tree-based attention mechanism to verify candidates efficiently. Achieves 2-3x speedup without needing a separate draft model, requiring only fine-tuning of the added heads.

- **Project angle:** Train Medusa heads for a LLaMA or Vicuna model, compare against EAGLE and standard speculative decoding, and analyze how the number of heads and tree structure affect speedup and quality.

3.5 Scissorhands: KV-Cache Compression via Importance Persistence

- **Title:** Scissorhands: Exploiting the Persistence of Importance Hypothesis for LLM KV Cache Compression at Test Time
- **Authors:** Liu, Z., Desai, A., Liao, F., Wang, W., Xie, V., Xu, Z., Kyrillidis, A., and Shrivastava, A.
- **Venue:** NeurIPS 2023
- **Description:** Discovers that tokens with high attention scores at one generation step tend to remain important at future steps ("persistence of importance"). Exploits this to evict unimportant KV-cache entries, reducing KV-cache memory by up to 5x with negligible quality loss.
- **Project angle:** Implement the importance-based KV-cache eviction policy on top of a HuggingFace LLM (LLaMA-2-7B), measure memory savings and generation quality on long-context benchmarks, and compare with H2O and StreamingLLM.

3.6 H2O: Heavy-Hitter Oracle for KV-Cache Efficiency

- **Title:** H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models
- **Authors:** Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Re, C., Barrett, C., Wang, Z., and Chen, B.
- **Venue:** NeurIPS 2023
- **Description:** Identifies "Heavy Hitter" tokens that receive disproportionately high attention scores across all layers and timesteps. Proposes retaining only these heavy hitters plus recent tokens in the KV-cache, achieving 20x compression with minimal quality degradation. Provides theoretical justification via a connection to the attention mechanism.
- **Project angle:** Implement H2O's KV-cache eviction on LLaMA-2-7B, compare with Scissorhands and full KV-cache on long-context tasks (summarization, multi-document QA), and visualize the heavy-hitter patterns across layers.

3.7 StreamingLLM: Efficient Infinite-Length Generation

- **Title:** Efficient Streaming Language Models with Attention Sinks
- **Authors:** Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M.
- **Venue:** ICLR 2024
- **Description:** Discovers "attention sinks" -- the first few tokens in a sequence that absorb disproportionately high attention regardless of content. Proposes keeping these attention sink tokens plus a sliding window of recent tokens in the KV-cache, enabling stable generation over millions of tokens with constant memory.
- **Project angle:** Deploy StreamingLLM on LLaMA-2-7B with different window sizes, evaluate on long-form generation and streaming tasks, and compare memory usage and quality against H2O and full-context inference.

3.8 vLLM: Paged KV-Cache Management

- **Title:** Efficient Memory Management for Large Language Model Serving with PagedAttention

- **Authors:** Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I.
- **Venue:** SOSP 2023
- **Description:** Proposes PagedAttention, inspired by virtual memory paging, to manage KV-cache memory in LLM serving. Eliminates memory fragmentation and enables flexible memory sharing across sequences, improving throughput by 2-4x. Forms the core of the widely-used vLLM serving system.
- **Project angle:** Deploy vLLM and benchmark its throughput vs. HuggingFace generate() under varying batch sizes, sequence lengths, and concurrent request rates. Study the impact of paged KV-cache management on serving latency and throughput. Code at github.com/vllm-project/vllm.

3.9 Mixtral of Experts: Sparse MoE for Efficient LLMs

- **Title:** Mixtral of Experts
- **Authors:** Jiang, A. Q., Sablayrolles, A., et al. (Mistral AI)
- **Venue:** arXiv 2024 (extremely high impact)
- **Description:** Introduces Mixtral 8x7B, a sparse Mixture of Experts model using 2 out of 8 experts per token, providing 47B total parameters but only 13B active parameters per forward pass. Matches or outperforms LLaMA-2-70B and GPT-3.5 on most benchmarks while being 6x faster at inference.
- **Project angle:** Analyze the MoE routing mechanism -- which experts specialize in what types of tokens? Benchmark inference speed vs. dense models of equivalent active parameters. Fine-tune Mixtral with LoRA and study how adapters interact with expert routing.

3.10 DeepSeek-V2: Multi-Head Latent Attention

- **Title:** DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model
- **Authors:** DeepSeek AI
- **Venue:** arXiv 2024 (extremely high impact)
- **Description:** Introduces Multi-head Latent Attention (MLA), which compresses KV-cache by projecting keys and values into a low-rank latent space, reducing per-token KV-cache from ~200 bytes to ~5 bytes while maintaining quality. Combined with DeepSeekMoE architecture, achieves significant efficiency gains over dense models.
- **Project angle:** Study and benchmark the MLA mechanism -- compare KV-cache memory savings against standard Multi-Head Attention and GQA (Grouped Query Attention) at different sequence lengths. Analyze the quality-efficiency trade-off of the low-rank KV compression.

Category 4: Efficient Training and Fine-Tuning (7 papers)

4.1 QLoRA: Quantized Low-Rank Adaptation

- **Title:** QLoRA: Efficient Finetuning of Quantized LLMs
- **Authors:** Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L.
- **Venue:** NeurIPS 2023
- **Description:** Combines 4-bit NormalFloat quantization of the base model with LoRA adapters, enabling fine-tuning of a 65B parameter model on a single 48GB GPU. Introduces NF4 data type, double quantization, and paged optimizers. The resulting Guanaco models match ChatGPT quality.

- **Project angle:** Fine-tune LLaMA-2-7B or LLaMA-3-8B with QLoRA using a single consumer GPU (24GB), compare NF4 vs. FP4 base quantization, evaluate on MMLU and MT-Bench, and measure training memory savings. Code via github.com/huggingface/peft.

4.2 DoRA: Weight-Decomposed Low-Rank Adaptation

- **Title:** DoRA: Weight-Decomposed Low-Rank Adaptation
- **Authors:** Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H.
- **Venue:** ICML 2024 (Oral)
- **Description:** Decomposes pre-trained weight into magnitude and direction components, then applies LoRA specifically to the directional component. This decomposition more closely mimics the learning behavior of full fine-tuning while maintaining LoRA's parameter efficiency. Consistently outperforms LoRA on LLaMA, LLaVA, and VL-BART with zero additional inference overhead.
- **Project angle:** Compare DoRA vs. LoRA vs. QLoRA on the same fine-tuning tasks (instruction following, domain adaptation), varying rank, and measure training cost, final quality, and convergence speed. Code at github.com/NVlabs/DoRA.

4.3 LongLoRA: Efficient Long-Context Fine-Tuning

- **Title:** LongLoRA: Efficient Fine-tuning of Long-Context Large Language Models
- **Authors:** Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., and Jia, J.
- **Venue:** ICLR 2024 (Oral)
- **Description:** Proposes Shifted Sparse Attention (S2-Attn) for efficient long-context fine-tuning, requiring only two lines of code change during training while supporting full attention at inference. Extends LLaMA-2-7B from 4K to 100K context length on a single 8xA100 machine.
- **Project angle:** Fine-tune a LLaMA model to extend its context window using LongLoRA, compare S2-Attn vs. full attention fine-tuning in terms of cost and quality, and evaluate on long-context benchmarks (SCROLLS, LongBench). Code at github.com/dvlab-research/LongLoRA.

4.4 GaLore: Memory-Efficient Training via Gradient Projection

- **Title:** GaLore: Memory-Efficient LLM Training by Gradient Low-Rank Projection
- **Authors:** Zhao, J., Zhang, Z., Chen, B., Wang, Z., Anandkumar, A., and Tian, Y.
- **Venue:** ICML 2024
- **Description:** Projects gradients into a low-rank subspace during training, enabling full-parameter learning (unlike LoRA which only learns low-rank updates) with up to 65% memory reduction in optimizer states. Enables pretraining a LLaMA-7B-equivalent model on a single 24GB GPU.
- **Project angle:** Apply GaLore to fine-tune or pretrain a transformer (125M-1B), compare memory usage and convergence against Adam+LoRA and standard Adam, and experiment with different projection ranks and update frequencies. Code at github.com/jiaweizzhao/GaLore.

4.5 Sophia: Scalable Second-Order Optimizer

- **Title:** Sophia: A Scalable Stochastic Second-Order Optimizer for Language Model Pre-Training
- **Authors:** Liu, H., Li, Z., Hall, D., Liang, P., and Ma, T.
- **Venue:** ICLR 2024
- **Description:** Proposes a lightweight second-order optimizer using a diagonal Hessian estimate clipped element-wise, achieving 2x speedup over Adam in total compute for pretraining LLMs to the

same loss. Uses curvature information to adaptively set per-parameter update sizes.
- **Project angle:** Implement Sophia and compare pretraining curves against AdamW for a GPT-2 scale model (125M-350M), analyze the Hessian estimates across layers, and measure wall-clock time to target perplexity.

4.6 LoftQ: LoRA-Aware Quantization

- **Title:** LoftQ: LoRA-Fine-Tuning-Aware Quantization for Large Language Models
- **Authors:** Li, Y., Yu, Y., Liang, C., He, P., Karampatziakis, N., Chen, W., and Zhao, T.
- **Venue:** ICLR 2024
- **Description:** Addresses the gap between quantization and LoRA fine-tuning by jointly optimizing the quantized backbone weights and LoRA initialization. Alternates between quantizing the weights and finding the best low-rank approximation of the quantization residual, significantly outperforming naive QLoRA on challenging tasks like code generation and math.
- **Project angle:** Compare LoftQ vs. QLoRA on fine-tuning LLaMA-2-7B at different bit-widths (2-bit, 4-bit) on diverse tasks (summarization, code, math), analyzing where the improved initialization matters most.

4.7 Scaling Data-Constrained Language Models

- **Title:** Scaling Data-Constrained Language Models
- **Authors:** Muennighoff, N., Rush, A. M., Barak, B., Le Scao, T., Tazi, N., Piktus, A., Pyysalo, S., Wolf, T., and Raffel, C.
- **Venue:** NeurIPS 2023
- **Description:** Systematically studies what happens when LLMs are trained with repeated data (up to 4 epochs), finding that repeating data is surprisingly effective -- training with 4 epochs of unique data matches the compute-optimal allocation of training with 1 epoch of 4x more data. Provides new scaling laws accounting for data repetition and data quality.
- **Project angle:** Reproduce the scaling law experiments at small scale (125M-350M models), validate whether data repetition scaling holds for your specific data mix, and explore the interaction between data quality filtering and repetition tolerance.

Category 5: Efficient Visual Generation (10 papers)

5.1 Consistency Models: Fast Few-Step Generation

- **Title:** Consistency Models
- **Authors:** Song, Y., Dhariwal, P., Chen, M., and Sutskever, I.
- **Venue:** ICML 2023
- **Description:** Proposes a new family of generative models that learn to map any point on a diffusion ODE trajectory directly to the trajectory's origin, enabling high-quality single-step or few-step generation. Can be trained by distillation from a pretrained diffusion model or from scratch.
- **Project angle:** Train a consistency model by distilling a pretrained DDPM on CIFAR-10, compare 1-step, 2-step, and 4-step generation quality (FID/IS) against DDPM 1000-step and DDIM 50-step sampling. Code at github.com/openai/consistency_models.

5.2 Improved Consistency Training (iCT)

- **Title:** Improved Techniques for Training Consistency Models
- **Authors:** Song, Y. and Dhariwal, P.
- **Venue:** ICLR 2024
- **Description:** Identifies and addresses key training instabilities in consistency models through improved noise schedules, loss weighting, and EMA strategies. Achieves FID 2.51 on CIFAR-10 and 3.25 on ImageNet 64x64 in two-step generation, closing the gap with diffusion models.
- **Project angle:** Implement the improved training techniques on CIFAR-10 consistency models, perform ablation studies of each proposed improvement, and measure FID progression during training.

5.3 Latent Consistency Models (LCM)

- **Title:** Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference
- **Authors:** Luo, S., Tan, Y., Huang, L., Li, J., and Zhao, H.
- **Venue:** arXiv 2023 (extremely high impact, widely deployed)
- **Description:** Extends consistency models to the latent space of Stable Diffusion, enabling high-quality 512x768 image generation in just 2-4 steps. Introduces Latent Consistency Distillation and the LCM-LoRA approach that can be applied to any fine-tuned SD model without retraining.
- **Project angle:** Apply LCM-LoRA to different Stable Diffusion checkpoints (SD 1.5, SDXL), benchmark FID/CLIP scores at 1/2/4/8 steps, and compare against SDXL-Turbo and DDIM sampling at various step counts.

5.4 DiT: Scalable Diffusion with Transformers

- **Title:** Scalable Diffusion Models with Transformers
- **Authors:** Peebles, W. and Xie, S.
- **Venue:** ICCV 2023
- **Description:** Replaces the standard U-Net backbone in diffusion models with a vision transformer (DiT), demonstrating that transformers are more scalable and achieve better FID than U-Nets when given sufficient compute. DiT-XL/2 achieves state-of-the-art FID 2.27 on ImageNet 256x256 class-conditional generation.
- **Project angle:** Train DiT models at different scales (S, B, L) on CIFAR-10 or ImageNet-64, compare scaling behavior against U-Net based diffusion, and profile GPU utilization and training efficiency. Code at github.com/facebookresearch/DiT.

5.5 PixArt-alpha: Efficient Diffusion Transformer Training

- **Title:** PixArt-alpha: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis
- **Authors:** Chen, J., Yu, J., Ge, C., Yao, L., Xie, E., Wu, Y., Wang, Z., Kwok, J., Luo, P., Lu, H., and Li, Z.
- **Venue:** ICLR 2024 (Spotlight)
- **Description:** Achieves competitive text-to-image quality with only 10.8% of Stable Diffusion v1.5's training cost (675 vs. 6,250 A100 GPU-days) by decomposing training into three stages: learning pixel distributions from a pretrained model, learning text-image alignment via a curated dataset, and high-aesthetic training on a filtered subset.
- **Project angle:** Reproduce the efficient training pipeline on a smaller scale, analyze the contribution of each training stage, and benchmark against direct training without the staged approach. Code at github.com/PixArt-alpha/PixArt-alpha.

5.6 Structural Pruning for Diffusion Models

- **Title:** Structural Pruning for Diffusion Models
- **Authors:** Fang, G., Ma, X., and Wang, X.
- **Venue:** NeurIPS 2023
- **Description:** First work to apply structural pruning to diffusion models, addressing the challenge that diffusion architectures contain residual and skip connections creating complex dependency structures. Achieves 2x speedup with minimal FID degradation on DDPM and Stable Diffusion.
- **Project angle:** Prune a pretrained DDPM or Stable Diffusion at various compression ratios, compare against naive magnitude pruning, and measure the FID-vs-speedup trade-off. Analyze which layers in the U-Net are most redundant.

5.7 VideoLDM: Efficient Latent Video Generation

- **Title:** Align Your Latents: High-Resolution Video Synthesis with Latent Diffusion Models
- **Authors:** Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Karras, T.
- **Venue:** CVPR 2023
- **Description:** Extends latent diffusion models to video generation by inserting temporal alignment layers into a pretrained image LDM, fine-tuning only these temporal layers on video data. This approach is far more efficient than training a video diffusion model from scratch, as it leverages pretrained image generation capabilities.
- **Project angle:** Study the temporal layer insertion strategy, fine-tune temporal layers on a small video dataset (e.g., UCF-101), and compare against naive frame-by-frame generation in terms of temporal consistency and FVD.

5.8 AnimateDiff: Modular Video Animation

- **Title:** AnimateDiff: Animate Your Personalized Text-to-Image Diffusion Models without Specific Tuning
- **Authors:** Guo, Y., Yang, C., Rao, A., Wang, Y., Qiao, Y., Lin, D., and Dai, B.
- **Venue:** ICLR 2024
- **Description:** Trains a plug-and-play motion module that can be inserted into any personalized text-to-image model (e.g., SD + LoRA) to animate it without model-specific fine-tuning. Decouples motion learning from appearance, making video generation efficient and modular.
- **Project angle:** Deploy AnimateDiff with different personalized SD models (DreamBooth, LoRA-based), evaluate temporal consistency and motion quality, and benchmark compute requirements compared to training a full video model.

5.9 SDXL-Turbo: Adversarial Diffusion Distillation

- **Title:** Adversarial Diffusion Distillation
- **Authors:** Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. (Stability AI)
- **Venue:** ECCV 2024
- **Description:** Combines score distillation with adversarial training to distill SDXL into a model that generates high-quality 512x512 images in just 1-4 steps. The adversarial loss provides high-frequency detail that score distillation alone misses, achieving the first real-time high-quality text-to-image generation.

- **Project angle:** Benchmark SDXL-Turbo at 1/2/4 steps against LCM and DDIM on SDXL, evaluate using FID, CLIP score, and human preference, and analyze the trade-off between adversarial training strength and output diversity.

5.10 DMD: One-Step Diffusion via Distribution Matching

- **Title:** One-step Diffusion with Distribution Matching Distillation
- **Authors:** Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T.
- **Venue:** CVPR 2024
- **Description:** Distills a diffusion model into a one-step generator by minimizing the KL divergence between the generator's output distribution and the diffusion model's distribution, using a regression loss for stability combined with a distribution matching loss for quality. Achieves state-of-the-art one-step generation quality on ImageNet.
- **Project angle:** Reproduce DMD distillation on a smaller diffusion model (CIFAR-10 or ImageNet-64), compare one-step quality against consistency models and progressive distillation, and analyze the role of the regression vs. distribution matching losses.

5.11 PTQ4DiT: Post-Training Quantization for Diffusion Transformers

- **Title:** PTQ4DiT: Post-training Quantization for Diffusion Transformers
- **Authors:** Wu, J., Li, S., Chen, Y., Shang, Y., and Abdi, A. H.
- **Venue:** NeurIPS 2024
- **Description:** First dedicated post-training quantization method for Diffusion Transformers (DiT). Proposes Channel-wise Salience Balancing (CSB) to redistribute extreme channel magnitudes and Spearman's rho-guided Salience Calibration (SSC) to handle temporal variations in activation distributions across denoising steps. Successfully quantizes DiT to W8A8 with comparable quality and enables W4A8 for the first time.
- **Project angle:** Apply PTQ4DiT to quantize a pretrained DiT or PixArt-alpha model at W8A8 and W4A8, compare FID against naive PTQ and full-precision, and analyze how quantization sensitivity varies across timesteps and layers. Code at github.com/adreamwu/PTQ4DiT.

5.12 EfficientDM: Quantization-Aware Fine-Tuning for Diffusion Models

- **Title:** EfficientDM: Efficient Quantization-Aware Fine-Tuning of Low-Bit Diffusion Models
- **Authors:** He, Y., Liu, J., Wu, W., Zhou, H., and Zhuang, B.
- **Venue:** ICLR 2024 (Spotlight)
- **Description:** Proposes a data-free, parameter-efficient QAT framework for diffusion models. Introduces QALoRA (quantization-aware LoRA) that can be merged with model weights and jointly quantized, scale-aware optimization to address cross-layer scale variation, and temporal learned step-size quantization for activation distributions that vary across denoising steps. Achieves only 0.05 sFID increase at 4-bit (W4A4) on LDM-4 ImageNet 256x256 with 16.2x faster quantization than standard QAT.
- **Project angle:** Apply EfficientDM to quantize Stable Diffusion or LDM to 4-bit, compare PTQ (Q-Diffusion) vs. QAT (EfficientDM) approaches, and analyze the contribution of each proposed component via ablation. Code at github.com/ThisisBillhe/EfficientDM.

5.13 SVDQuant: Low-Rank-Assisted 4-Bit Diffusion Models

- **Title:** SVDQuant: Absorbing Outliers by Low-Rank Components for 4-Bit Diffusion Models
- **Authors:** Li, M., Lin, J., Tang, H., and Han, S. (MIT Han Lab)
- **Venue:** ICLR 2025 (Spotlight)
- **Description:** Proposes a novel PTQ paradigm that absorbs weight and activation outliers into a low-rank SVD branch rather than smoothing them. Co-designs the Nunchaku inference engine that fuses the low-rank branch kernels into the low-bit branch to avoid memory overhead. Reduces memory of the 12B FLUX.1 model by 3.5x and achieves 3.0x speedup over W4A16 baseline on an RTX 4090 laptop GPU.
- **Project angle:** Apply SVDQuant to quantize FLUX.1 or Stable Diffusion 3 to 4-bit, compare against Q-Diffusion and PTQ4DiT, benchmark quality (FID, CLIP score) and actual inference speedup, and analyze the learned low-rank components. Code at github.com/mit-han-lab/nunchaku.

5.14 DeepCache: Training-Free Diffusion Acceleration via Feature Caching

- **Title:** DeepCache: Accelerating Diffusion Models for Free
- **Authors:** Ma, X., Fang, G., and Wang, X.
- **Venue:** CVPR 2024
- **Description:** Exploits temporal redundancy in sequential denoising steps by caching and reusing high-level features from the U-Net across adjacent timesteps. Achieves 2.3x speedup on Stable Diffusion v1.5 with only 0.05 CLIP Score decrease, and 4.1x speedup on LDM-4 (ImageNet) with 0.22 FID increase -- all without any training or fine-tuning.
- **Project angle:** Apply DeepCache to accelerate Stable Diffusion or DDPM, experiment with different caching intervals and layers, compare against DDIM (fewer steps) and LCM (distilled fewer-step model), and analyze which features are most cacheable. Code at github.com/horseee/DeepCache.

5.15 Learning-to-Cache: Dynamic Layer Caching for Diffusion Transformers

- **Title:** Learning-to-Cache: Accelerating Diffusion Transformer via Layer Caching
- **Authors:** Ma, X., Fang, G., Mi, M. B., and Wang, X.
- **Venue:** NeurIPS 2024
- **Description:** Extends the caching concept to Diffusion Transformers (DiT) with a learned, dynamic caching policy. Trains a lightweight router that decides which layers to skip at each timestep, exploiting the identical layer structure of transformers. Removes up to 93.68% of computation in cache steps (46.84% overall) on U-ViT-H/2 with less than 0.01 FID drop.
- **Project angle:** Apply Learning-to-Cache to a pretrained DiT model, compare against DeepCache and uniform step-skipping baselines, analyze the learned caching patterns (which layers are skipped at which timesteps), and benchmark actual wall-clock speedup. Code at github.com/horseee/learning-to-cache.

Category 6: On-Device and Edge AI (5 papers)

6.1 MLC-LLM: Universal On-Device LLM Deployment

- **Title:** MLC-LLM: Universal LLM Deployment Engine with ML Compilation
- **Authors:** Team MLC (Chen, T., et al.)
- **Venue:** Open-source project / Technical report, 2023--2024

- **Description:** Provides a universal deployment solution for LLMs on diverse hardware (iPhone, Android, GPU, WebGPU) using machine learning compilation (Apache TVM). Enables running LLaMA-7B on an iPhone at interactive speeds through optimized quantization, operator fusion, and memory planning.
- **Project angle:** Deploy a quantized LLM (e.g., LLaMA-3.2-3B, Phi-3-mini) on a laptop or phone using MLC-LLM, benchmark token throughput at different quantization levels (4-bit vs. 3-bit), and compare with llama.cpp. Code at github.com/mlc-ai/mlc-llm.

6.2 MobileDiffusion: Instant Text-to-Image on Mobile

- **Title:** MobileDiffusion: Instant Text-to-Image Generation on Mobile Devices
- **Authors:** Zhao, Y., Xu, Y., Xiao, Z., Jia, H., and Hou, T. (Google)
- **Venue:** arXiv 2024 (highly cited)
- **Description:** Designs an efficient UNet architecture optimized for mobile text-to-image generation, combined with distillation to generate 512x512 images in under 0.5 seconds on an iPhone 15 Pro. Uses architecture-specific optimizations (depthwise separable convolutions, efficient attention) and step distillation.
- **Project angle:** Study the mobile-specific architecture choices, benchmark against standard Stable Diffusion (with quantization) on mobile, and analyze the latency-quality trade-off of different components (UNet efficiency vs. step distillation).

6.3 PowerInfer: Consumer-GPU LLM Serving

- **Title:** PowerInfer: Fast Large Language Model Serving with a Consumer-grade GPU
- **Authors:** Xue, Y., et al. (SJTU IPADS)
- **Venue:** SOSP 2024
- **Description:** Exploits the locality in LLM inference -- only a small set of neurons are "hot" (frequently activated) for any given input. Pre-loads hot neurons on the GPU while computing cold neurons on the CPU, achieving up to 11x speedup over llama.cpp on a single RTX 4090 for 175B-parameter models.
- **Project angle:** Deploy PowerInfer on a consumer GPU with an OPT or LLaMA model, profile the hot/cold neuron distribution, compare inference speed with llama.cpp and vLLM, and analyze how the hot neuron set varies across different input types. Code at github.com/SJTU-IPADS/PowerInfer.

6.4 Phi-3: Strong Small Language Models

- **Title:** Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone
- **Authors:** Abdin, M., et al. (Microsoft Research)
- **Venue:** arXiv 2024 (extremely high impact)
- **Description:** Demonstrates that a 3.8B parameter model (Phi-3-mini), when trained on carefully curated "textbook-quality" data, achieves performance comparable to Mixtral 8x7B and GPT-3.5 on many benchmarks. Runs natively on an iPhone 14 at 12 tokens/sec, proving that data quality can dramatically compensate for model size.
- **Project angle:** Benchmark Phi-3-mini against larger models (LLaMA-2-7B, Mistral-7B) on diverse tasks, deploy on a mobile device using MLC-LLM or llama.cpp, and study how QLoRA fine-tuning interacts with the already data-efficient training.

6.5 LLaMA 3.2: Lightweight Frontier Models

- **Title:** LLaMA 3.2: Lightweight Text and Multimodal Models
- **Authors:** Meta AI
- **Venue:** Meta Release, September 2024
- **Description:** Releases 1B and 3B parameter language models that achieve strong performance for their size class, specifically designed for on-device deployment. The 3B model approaches the quality of the original LLaMA-2-7B while being more than 2x smaller. Also includes 11B and 90B vision-language models with efficient visual token processing.
- **Project angle:** Benchmark LLaMA-3.2-1B and 3B against Phi-3-mini and other small models on standard benchmarks, deploy on consumer hardware with quantization (4-bit, 2-bit), and evaluate the accuracy-latency Pareto frontier for on-device use cases.

Summary Statistics

Category	Count
1. LLM Pruning and Knowledge Distillation	8
2. Quantization	9
3. Efficient LLM Inference and Serving	10
4. Efficient Training and Fine-Tuning	7
5. Efficient Visual Generation	15
6. On-Device and Edge AI	5
Total	**54**