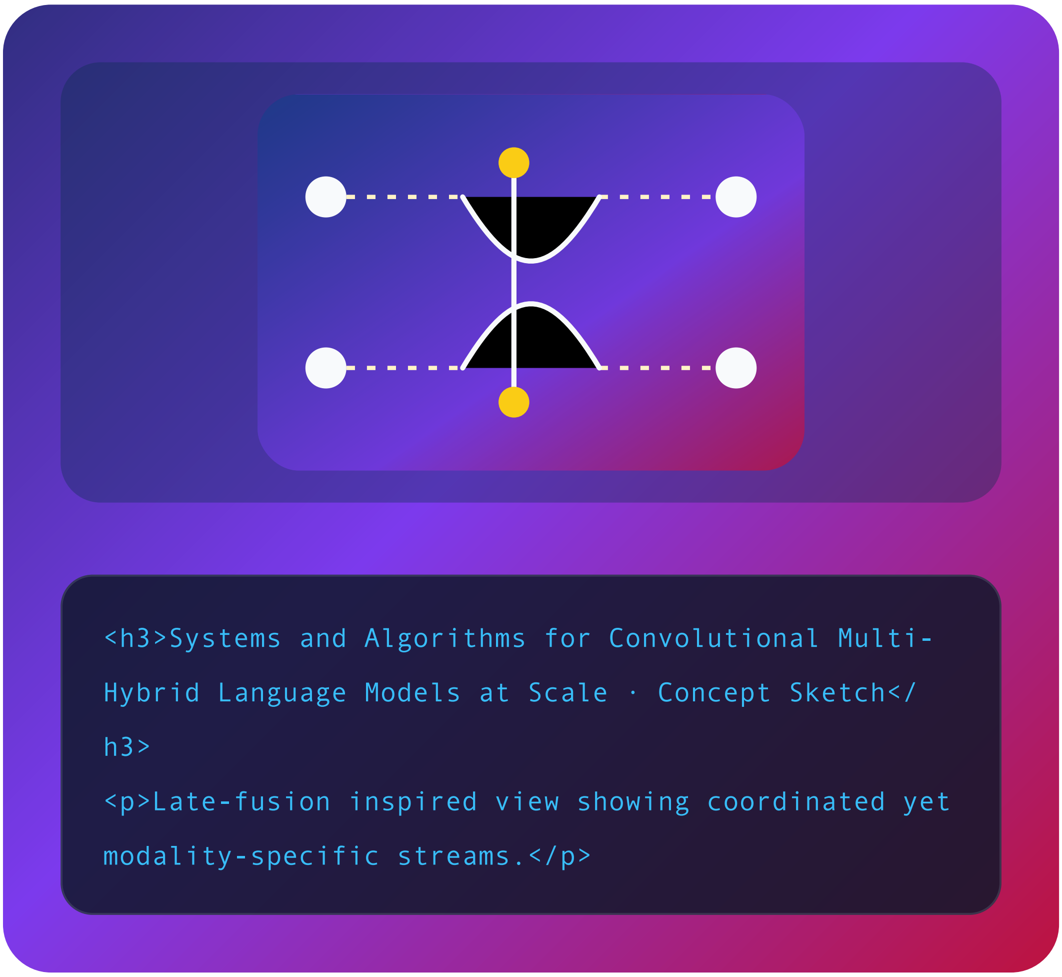

```
<h3>Systems and Algorithms for Convolutional Multi-
Hybrid Language Models at Scale · Concept Sketch</
h3>
<p>Late-fusion inspired view showing coordinated yet
modality-specific streams.</p>
```

Systems and Algorithms for Convolutional Multi-Hybrid Language Models at Scale

Authors: Jerome Ku, Eric Nguyen, David W. Romero, Garyk Brixi, Brandon Yang, Anton Vorontsov, Ali Taghibakhshi, Amy X. Lu, Dave P. Burke, Greg Brockman, Stefano Massaroli, Christopher Ré, Patrick D. Hsu, Brian L. Hie, Stefano Ermon, Michael Poli

Year: 2025

Venue: arXiv preprint

1. Classification

- **Domain Category:**
  - **General FM survey / theory (systems-focused).** This paper focuses on systems and algorithms for training multi-hybrid language models at scale, with applications demonstrated on genomics (Evo 2) but principles applicable to any sequence modeling domain.
- **FM Usage Type:**
  - **Core FM development (systems/architecture).** The paper introduces StripedHyena 2, a convolutional multi-hybrid architecture, and provides systems-level optimizations (kernels, parallelism) for efficient training at 40B parameter scale.
- **Key Modalities:**
  - Primarily focused on byte-tokenized sequences (demonstrated on DNA/nucleotide sequences in Evo 2), but principles apply to any sequential data (text, audio, time series).

2. Executive Summary

This paper introduces convolutional multi-hybrid architectures—a new class of sequence models that combine multiple types of input-dependent convolutional operators with complementary capabilities—and provides the systems and algorithms needed to train them efficiently at scale. The key insight is that different operators excel at different subtasks: input-dependent convolutions are good at multi-token recall and filtering (useful for byte-level data), while attention is optimized for targeted recall across longer sequences. Rather than having multiple operators compete for the same capability (in-context recall), multi-hybrids specialize operators to complementary roles. The paper focuses on StripedHyena 2, which combines three types of Hyena operators: Hyena-SE (short explicit filters, hardware-optimized for local multi-token recall), Hyena-MR (medium regularized filters for hundreds of tokens), and Hyena-LI (long implicit filters for entire sequence). At the 40 billion parameter scale, StripedHyena 2 trains 1.2-2.9× faster than optimized Transformers and 1.1-1.4× faster than previous generation hybrids. The paper provides detailed systems contributions: overlap-add blocked kernels for tensor cores that achieve 2× throughput improvement over linear attention and state-space models, and custom context parallelism strategies (all-to-all and point-to-point) for distributed training of long sequences. Effectiveness is demonstrated through Evo 2, a state-of-the-art 40B parameter foundation model for genomics trained on 9 trillion tokens with 1 million context length. This work demonstrates the importance of architecture-hardware co-design, showing how algorithmic innovations (multi-hybrid operators) combined with systems optimizations (efficient kernels, parallelism) enable practical training of large-scale foundation models that outperform Transformers in both quality and efficiency.

3. Problem Setup and Motivation

- **Scientific / practical problem**
  - Transformers dominate language modeling but face efficiency challenges: quadratic attention scaling limits context length, and fixed-state operators (linear attention, SSMs) only realize efficiency gains at very long sequences where they underperform full attention.
  - Hybrid architectures (combining multiple operators) have been proposed but often introduce redundancy (multiple operators optimized for the same capability) and struggle to surpass Transformers in common pretraining regimes (shorter contexts, larger models).
  - The goal is to design architectures that are both hybridization-aware (operators with complementary capabilities) and hardware-

aware (efficient implementations), enabling better quality and efficiency across a range of input and model sizes.

- **Why this is hard**
  - **Operator specialization:**
    - Need to identify which operators excel at which subtasks (recall, compression, multi-token recall, fuzzy recall).
    - Must design architectures that leverage complementary strengths rather than redundant capabilities.
  - **Hardware efficiency:**
    - Convolutional operators need efficient implementations on modern GPUs (tensor cores).
    - Standard convolution libraries may not be optimized for the specific filter lengths and patterns used in multi-hybrids.
  - **Distributed training:**
    - Long sequences require context parallelism (splitting sequences across devices).
    - Different operator types (short explicit, medium regularized, long implicit) need different parallelism strategies.
  - **Scaling validation:**
    - Need to demonstrate improvements at billion-parameter scale, not just small models.
  - **Architecture-hardware co-design:**
    - Must simultaneously optimize architecture (which operators, how to compose) and systems (kernels, parallelism) for best results.

4. Data and Modalities

- **Datasets used**
  - **Evo 2 pretraining:**
    - OpenGenome2: 8.84-9.3 trillion nucleotides from 850+ species (prokaryotes, eukaryotes, organelles, metagenomic).
    - Context length: up to 1 million tokens (nucleotides).
  - **Evaluation:**
    - Evo 2 performance on genomic tasks (mutational effect prediction, variant effect prediction, sequence generation).
    - Scaling experiments comparing Transformers, multi-hybrids, and other operators.
- **Modalities**
  - **Primary:** DNA sequence (byte-tokenized nucleotides).
  - **Principles apply to:** Any sequential data (text, audio, time series, code).
- **Preprocessing / representation**
  - **Byte tokenization:** Each nucleotide (A, C, G, T) is a byte token.
  - **Sequence packing:** Efficient batching of variable-length sequences.
  - **Context length:** Up to 1 million tokens during training.

5. Model / Foundation Model

- **Model Type**
  - **Convolutional multi-hybrid architecture** (StripedHyena 2) combining multiple Hyena operator variants in a “striped” pattern.
  - Based on input-dependent convolutions with data-controlled gating.

- **Is it a new FM or an existing one?**
  - **New architecture class.** StripedHyena 2 is a new multi-hybrid architecture, demonstrated through Evo 2 (a complete foundation model), but the paper focuses on the systems and algorithms rather than the full FM.
- **Key components and innovations**
  - **Three Hyena operator types:**
    - **Hyena-SE (Short Explicit):** Short, explicitly parameterized filters (length 4-7). Hardware-optimized for local multi-token recall. Highest throughput of any sequence mixing operator.
    - **Hyena-MR (Medium Regularized):** Explicitly parameterized filters of length hundreds, with exponential decay regularizer. Efficient modeling across hundreds of tokens.
    - **Hyena-LI (Long Implicit):** Long filters parameterized implicitly via linear combination of real exponentials. Aggregates information over entire sequence. Can switch to recurrent parametrization for constant memory.
  - **Striped composition:**
    - Operators arranged in “striped” pattern: SE-MR-LI, SE-SE-LI, etc.
    - Different patterns for different layers, enabling specialization.
  - **Filter grouping:**
    - Groups channels share filters, improving hardware utilization and enabling efficient blocked kernels.
  - **Systems optimizations:**
    - **Two-stage blocked kernels:** Overlap-add algorithms adapted to tensor cores for Hyena-SE and Hyena-MR.
    - **Context parallelism:** Custom all-to-all and point-to-point strategies for distributed training of long sequences.
- **Training setup**
  - **Scale:** 40 billion parameters, 9 trillion tokens, 1 million context length (Evo 2).

    - **Efficiency:** 1.2-2.9× faster training than optimized Transformers, 1.1-1.4× faster than previous hybrids.
    - **Hardware:** H100 GPUs, optimized kernels for tensor cores.
    - **Parallelism:** Context parallelism for sequences, standard data/tensor/pipeline parallelism for model.

6. Multimodal / Integration Aspects (If Applicable)

- **Not primarily multimodal:**
    - StripedHyena 2 is designed for sequential data (demonstrated on DNA sequences).
    - However, the architectural principles (multi-hybrid operators, efficient kernels) could potentially be extended to multimodal settings (e.g., integrating with vision encoders, as in some VLMs).
- **Potential extensions:**
    - Multi-hybrid architectures could combine sequence operators with cross-modal attention for vision-language or other multimodal tasks.
    - Systems optimizations (context parallelism, efficient kernels) are applicable to any long-sequence modeling, including multimodal sequences.

7. Experiments and Results

Main findings

- **Training efficiency at scale:**
    - StripedHyena 2 trains 1.2-2.9× faster than optimized Transformers at 40B scale.
    - 1.1-1.4× faster than previous generation hybrids.
    - Individual operators achieve 2× throughput improvement over linear attention and state-space models on H100 GPUs.
- **Evo 2 performance:**
    - State-of-the-art foundation model for genomics at 40B parameters.
    - Trained on 9 trillion tokens with 1 million context length.
    - Strong performance on mutational effect prediction, variant effect prediction, sequence generation.
- **Operator efficiency:**
    - **Hyena-SE:** Highest throughput of any sequence mixing operator, especially at short-medium sequences.
    - **Hyena-MR:** Efficient for hundreds of tokens, good balance of quality and speed.
    - **Hyena-LI:** Enables very long contexts (1M tokens) with sub-quadratic scaling.
- **Kernel performance:**
    - Two-stage blocked kernels achieve substantial speedups over PyTorch convolutions and FFT-based methods.
    - Tensor core utilization maximized through filter grouping and blocked algorithms.
- **Context parallelism:**
    - Custom all-to-all and point-to-point strategies enable efficient distributed training of 1M token sequences.
    - Channel-pipelined variants hide communication latency.

Ablation studies

- **Operator composition:**
    - Different striped patterns (SE-MR-LI vs. SE-SE-LI) affect performance and efficiency.
    - Optimal composition depends on sequence length and task.
- **Filter grouping:**
    - Grouping channels to share filters improves hardware utilization and enables efficient kernels.
- **Kernel implementations:**
    - Two-stage blocked kernels outperform naive PyTorch convolutions and FFT methods for short-medium filters.

Key insights

- **Architecture-hardware co-design is crucial:**
    - Designing operators (Hyena-SE, -MR, -LI) with hardware in mind (tensor cores, filter grouping) enables substantial efficiency gains.
- **Operator specialization works:**
    - Having operators with complementary capabilities (local recall, medium-range, long-range) is more effective than redundancy.
- **Systems optimizations matter:**
    - Custom kernels and parallelism strategies are essential for realizing architectural benefits at scale.

8. Strengths and Limitations

Strengths

- **Comprehensive systems work:**
    - Provides architecture design, kernel implementations, and parallelism strategies as a complete package.

- **Validated at scale:**
  - Demonstrates improvements at 40B parameter scale, not just small models.
- **Hardware-aware design:**
  - Operators and kernels designed for modern GPUs (tensor cores), maximizing efficiency.
- **Practical impact:**
  - Enables training of Evo 2, a state-of-the-art genomics foundation model.
- **General principles:**
  - While demonstrated on genomics, principles apply to any sequence modeling domain.

Limitations

- **Genomics-focused demonstration:**
  - While principles are general, main validation is on DNA sequences (Evo 2).
  - Performance on other domains (text, audio) not extensively shown.
- **Complexity:**
  - Multi-hybrid architectures with multiple operator types add complexity compared to simple Transformers.
- **Hyperparameter sensitivity:**
  - Optimal operator composition and filter lengths may need tuning per domain.
- **Limited comparison:**
  - While compared to Transformers and previous hybrids, comparison to other recent architectures (Mamba, xLSTM) is limited.
- **Systems expertise required:**
  - Implementing custom kernels and parallelism strategies requires significant systems engineering.

Open questions / future directions

- **Other domains:**
  - How do multi-hybrids perform on text, audio, or other sequential data?
- **Further optimizations:**
  - Can additional systems optimizations (quantization, sparsity) further improve efficiency?
- **Architecture search:**
  - Can we automatically discover optimal operator compositions for different tasks?
- **Integration with other techniques:**
  - How do multi-hybrids combine with MoE, quantization, or other efficiency techniques?
- **Theoretical understanding:**
  - Why do different operators excel at different subtasks? Can we formalize this?

9. Context and Broader Impact

Relation to other work

- **Compared to Transformers:**
  - Multi-hybrids achieve better quality and efficiency than Transformers at scale, especially for byte-tokenized data.
  - Sub-quadratic scaling enables longer contexts (1M tokens) that are infeasible with attention.
- **Compared to previous hybrids:**
  - Earlier hybrids (StripedHyena 1, Jamba) often had redundant operators.

- Multi-hybrids specialize operators to complementary capabilities, improving efficiency.

- **Compared to state-space models (Mamba, etc.):**
  - Multi-hybrids combine multiple operator types rather than using a single SSM.
  - Achieve better efficiency and quality across a range of sequence lengths.
- **Connection to Evo 2:**
  - Evo 2 demonstrates StripedHyena 2’s effectiveness as a complete foundation model for genomics.
  - Shows that systems optimizations enable practical training of 40B parameter models on 9T tokens.

Broader scientific and practical impact

- **Enables new scale:**
  - Systems optimizations make it practical to train 40B parameter models with 1M context, opening new applications.
- **Demonstrates architecture-hardware co-design:**
  - Shows how designing architectures with hardware in mind (tensor cores, parallelism) yields substantial gains.
- **Influences future work:**
  - Principles of operator specialization and systems optimization will influence future foundation model development.
- **Practical tools:**
  - Provides kernels and parallelism strategies that can be reused in other projects.

Open questions for future research

- **How to generalize further?**
  - Can multi-hybrid principles be extended to other domains (vision, audio, multimodal)?

- **Automated architecture search:**
    - Can we automatically discover optimal operator compositions for different tasks and hardware?
- **Further efficiency:**
    - What additional optimizations (quantization, sparsity, distillation) can further improve efficiency?
- **Theoretical foundations:**
    - Can we formalize why different operators excel at different subtasks?

10. Key Takeaways

1. **Architecture-hardware co-design is essential:**
   Designing operators and kernels together (rather than separately) enables substantial efficiency gains. Think about hardware (tensor cores, memory hierarchy) when designing architectures.

2. **Operator specialization beats redundancy:**
   Rather than having multiple operators compete for the same capability (in-context recall), specialize them to complementary roles (local recall, medium-range, long-range).

3. **Systems optimizations unlock scale:**
   Custom kernels (two-stage blocked algorithms) and parallelism strategies (context parallelism) are essential for training large models efficiently. Architecture alone isn’t enough.

4. **Multi-hybrids combine strengths:**
   Combining short explicit (hardware-optimized), medium regularized (efficient mid-range), and long implicit (global context) operators enables better quality and efficiency than any single operator type.

5. **Filter grouping improves utilization:**
   Having channels share filters enables efficient blocked kernels that maximize tensor core utilization.

6. **Context parallelism is crucial for long sequences:**
   For 1M token sequences, standard parallelism isn't enough; need custom context parallelism strategies (all-to-all, point-to-point).

7. **Validated at real scale:**
   Improvements demonstrated at 40B parameters, 9T tokens, not just small models. This is essential for practical impact.

8. **General principles, specific demonstration:**
   While demonstrated on genomics (Evo 2), principles of multi-hybrid architectures and systems optimization apply to any sequence modeling domain.

9. **Full-stack development matters:**
   Don't just propose an architecture; provide kernels, parallelism strategies, and training recipes as a complete package.

10. **This enables new capabilities:**
    Systems optimizations make it practical to train models with 1M context, enabling applications (whole-genome modeling, long-document understanding) that were previously infeasible.