Adaptive Neural Compression Engine

for Real-Time Streaming Video on Bandwidth-Constrained Networks

App. No. 18/472,839
Filed: March 14, 2024
Conf. No. 7821

Inventors Dr. Elena V. Sorokina (San Jose, CA); James A. Whitfield (Austin, TX); Priya Nair-Menon (Seattle, WA)

Assignee NeuroStream Technologies, Inc., 1250 Halcyon Drive, San Jose, CA 95134

Attorney/Agent Michael T. Hargreaves, Reg. No. 62,441; Meridian IP Group LLP

U.S. Class H04N 19/42; G06N 3/08; G06T 9/00

Filed March 14, 2024 (Priority: PCT/US2023/074481, Oct. 19, 2023)

Drawings 11 sheets, 14 figures

Cross-Reference to Related Applications

This application claims priority to U.S. Provisional Application No. 63/524,017, filed June 28, 2023, and is a continuation-in-part of U.S. Application No. 17/901,554, filed September 1, 2022, the entirety of each of which is incorporated herein by reference.

Background of the Invention

Field of the Invention

The present invention relates to video compression systems and, more particularly, to neural-network-driven adaptive bitrate encoding systems that dynamically reconfigure codec parameters in response to real-time network telemetry.

Description of the Related Art

Conventional adaptive bitrate (ABR) streaming systems, such as those conforming to *Dynamic Adaptive Streaming over HTTP* (DASH) and Apex HLS, partition encoded video into fixed-duration segments and select the next segment's quality level based on measured download throughput **(200)**. While effective under stable network conditions, such

heuristic-ladder approaches exhibit two principal shortcomings:

(i) **Lag in quality switching:** Rule-based bitrate ladders require multiple bandwidth samples before upshifting quality, introducing perceptible quality oscillations that degrade the *Mean Opinion Score* (MOS) by up to 1.8 points on ITU-T P.910 five-point scales.

(ii) **Scene-agnostic encoding:** Fixed quantization parameter (QP) schedules are blind to scene complexity and motion vectors, over-allocating bits to low-complexity scenes while under-serving fast-motion sequences.

Prior art disclosures, including U.S. Patent No. 10,484,727 (Zhao et al.) and U.S. Patent Application Publication No. US 2021/0160527 A1 (Gupta et al.), describe reinforcement-learning agents for bitrate selection but do not address real-time in-encoder parameter reconfiguration at frame granularity, nor do they provide a hardware-agnostic deployment substrate compatible with mobile system-on-chip (SoC) architectures.

There exists, therefore, a need in the art for an adaptive compression engine that (a) operates at intra-frame temporal resolution, (b) infers scene complexity jointly with network state, and (c) deploys on heterogeneous hardware without modification.

Summary of the Invention

The present invention provides an *Adaptive Neural Compression Engine* (ANCE) comprising a lightweight convolutional inference module, a predictive network telemetry interface, and a hardware abstraction layer enabling deterministic real-time operation on general-purpose processors, graphics processing units, and dedicated neural processing units.

In one aspect, the invention provides a computer-implemented method comprising: receiving, by a frame analysis module, a raw video frame and associated motion vectors from an upstream video capture pipeline; generating, by a convolutional neural network having fewer than 2.5 million parameters, a complexity score vector for each macroblock partition of said frame; receiving network bandwidth estimates and round-trip latency measurements from a telemetry aggregator; and outputting, by a decision engine, per-macroblock quantization parameters and an updated encoding profile within a latency budget not exceeding 4 milliseconds on a 1 GHz single-core processor.

In a further aspect, the invention provides a non-transitory computer-readable medium storing instructions that, when executed, perform the method described herein, as well as a hardware apparatus incorporating the foregoing method as a dedicated integrated circuit module.

Brief Description of the Drawings

FIG. 1 is a high-level block diagram of the Adaptive Neural Compression Engine (ANCE) system architecture **(100)**.

FIG. 2 is a data-flow diagram illustrating the frame-analysis pipeline **(110)**.

FIG. 3 is a schematic of the lightweight convolutional inference module (LCIM) **(120)**, showing input tensor dimensions and kernel sizes.

FIG. 4A is a graph depicting per-frame quantization parameter assignments under low-bandwidth network conditions (BW < 1.5 Mbps).

FIG. 4B is a graph depicting per-frame quantization parameter assignments under high-bandwidth network conditions (BW > 8 Mbps).

FIG. 5 is a flowchart of the decision engine inference loop **(130)**.

FIG. 6 is a block diagram of the hardware abstraction layer (HAL) **(140)** and its interfaces to heterogeneous processing units.

FIG. 7 is a sequence diagram showing interaction between the ANCE and a DASH manifest server during a live streaming session.

FIG. 8 is a graph comparing Mean Opinion Score (MOS) over time for the inventive ANCE versus three prior-art baseline systems.

FIG. 9 is a diagram of the telemetry aggregator data structures **(150)** and a corresponding ring-buffer memory layout.

FIG. 10 is a block diagram illustrating deployment of ANCE on a mobile SoC incorporating an ARM Cortex-A78 core and a dedicated NPU.

FIG. 11 is a diagram of the training pipeline for the convolutional inference module, including dataset provenance.

Detailed Description of the Preferred Embodiments

The following description is presented to enable any person skilled in the art to make and use the invention. Details are set forth for the purpose of explanation and are not to be construed as limiting. The scope of the invention is defined solely by the appended claims.

1. System Overview

Referring to FIG. 1 **(100)**, the Adaptive Neural Compression Engine (ANCE) comprises five principal subsystems arranged in a real-time processing pipeline:

1. *Frame Analysis Module (FAM)* **(110)** — accepts raw YCbCr 4:2:0 frames at up to 120 fps and computes block-level motion vector fields using a hierarchical block-matching

algorithm.

2. *Lightweight Convolutional Inference Module (LCIM)* **(120)** — a depthwise-separable convolutional network with six convolutional layers, two squeeze-and-excitation blocks, and a final regression head outputting a 32-dimensional complexity score vector per macroblock.

3. *Telemetry Aggregator (TA)* **(150)** — collects RTCP receiver reports and custom bandwidth probes via a ring buffer of 512 timestamped measurements, maintaining a Kalman-filtered estimate of available bandwidth.

4. *Decision Engine (DE)* **(130)** — a multi-layer perceptron (3 layers, 64 hidden units each) that jointly processes LCIM complexity vectors and TA bandwidth estimates to emit per-macroblock QP values and encoding profiles.

5. *Hardware Abstraction Layer (HAL)* **(140)** — provides a unified application programming interface (API) to dispatch inference workloads to a CPU, GPU (via OpenCL 3.0), or NPU (via NNAPI / CoreML delegate), selecting the fastest available execution context at runtime.

2. Frame Analysis Module

The FAM accepts frames via a shared-memory zero-copy interface. Luma (Y) and chroma (Cb, Cr) planes are processed separately. For each 16×16-pixel macroblock partition, the FAM computes a motion vector using a three-step hierarchical search with a maximum search range of ±32 pixels. Resulting motion magnitude m_i for macroblock i is defined by Equation (1):

$$m_i = \sqrt{\Delta x_i^2 + \Delta y_i^2} \tag{1}$$

where $(\Delta x_i,\ \Delta y_i)$ are the horizontal and vertical displacement components of the dominant motion vector for macroblock i. Motion magnitudes and raw luma values are concatenated into a $C \times H \times W$ input tensor delivered to the LCIM.

3. Lightweight Convolutional Inference Module (LCIM)

The LCIM is illustrated in FIG. 3 **(120)**. The network architecture is designed to fit within a 2.5-million-parameter budget, enabling real-time operation on low-power embedded processors. Layer specifications are as follows:

Layer	Configuration
Conv-1	3×3 depthwise, 32 channels, stride 1, ReLU6
Conv-2	1×1 pointwise, 64 channels, BN, ReLU6
SE-Block-1	Squeeze ratio $r = 4$; channel attention over 64-dim feature map
Conv-3	3×3 depthwise, 64 channels, stride 2, ReLU6
Conv-4	1×1 pointwise, 128 channels, BN, ReLU6
SE-Block-2	Squeeze ratio $r = 4$; 128-dim channel attention
Conv-5–6	1×1 pointwise, 64 then 32 channels, ReLU6
Regression Head	Global average pool → FC(32), linear activation

Model weights are stored in IEEE 754 half-precision (FP16) format; quantization to INT8 is optionally applied via post-training quantization (PTQ) with a representative calibration dataset of 5,000 frames drawn from the *UVG-HD* video benchmark.

4. Telemetry Aggregator and Kalman Filter

The TA maintains a circular buffer $\mathcal{B}$ of 512 bandwidth samples $\{b_t\}$ timestamped at millisecond resolution. A scalar Kalman filter estimates true available bandwidth $\hat{b}_t$ according to:

$$\hat{b}_{t|t-1} = \hat{b}_{t-1} \tag{2}$$

$$K_t = \frac{P_{t|t-1}}{P_{t|t-1} + R} \tag{3}$$

$$\hat{b}_t = \hat{b}_{t|t-1} + K_t\left(b_t - \hat{b}_{t|t-1}\right) \tag{4}$$

where R is the measurement noise variance, set by default to $R = 0.04\,\text{Mbps}^2$, and $P_{t|t-1}$ is the prior error covariance. Round-trip latency ℓ_t is estimated via exponentially weighted moving average with decay constant $\alpha = 0.1$.

5. Decision Engine

The DE receives a concatenated input vector $\mathbf{v} = [\mathbf{c}_i;\, \hat{b}_t;\, \ell_t;\, q_{t-1}]$ where $\mathbf{c}_i$ is the 32-dimensional LCIM complexity output for macroblock i, and q_{t-1} is the previously applied QP. The DE outputs a QP value $q_i \in [1, 51]$ and an encoding profile index $p \in \{0, 1, 2, 3\}$ corresponding to {baseline, main, high, high-10} H.264/AVC profiles. The DE is implemented as a 3-layer fully connected network with 64 hidden units, sigmoid-linear activation, and a 35-parameter output head.

6. Hardware Abstraction Layer

The HAL exposes three dispatch targets through a single `ance_infer()` function call. At initialization, the HAL benchmarks available backends using a 50-frame warm-up sequence and selects the backend whose mean latency per frame satisfies the 4 ms budget. If no backend meets the budget, the HAL falls back to a lightweight rule-based QP-selection algorithm described in § 5.7.

7. Fallback Rule-Based QP Selection

In the event that no backend meets the 4 ms latency budget, the ANCE activates a deterministic fallback: QP is set to $q_{\mathrm{fb}} = \mathrm{clamp}(51 - \lfloor 43 \cdot \hat{b}_t / b_{\max} \rfloor, 18, 42)$, where $b_{\max}$ is the peak bandwidth observed during the current session.

8. Training Methodology

The LCIM was trained on a corpus of 240,000 video clips (totalling 1.8 billion frames) drawn from the following sources: (i) YouTube-UGC dataset (ugc-dataset.github.io); (ii) BVI-Texture benchmark; and (iii) proprietary HDR mobile capture data licensed from ContentShield Inc. under a research agreement. Training employed the Adam optimizer ($\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) with a cosine annealing schedule over 120 epochs on 8× Quanta A100 GPUs.

Drawing Sheet — FIG. 1: System Architecture (100)

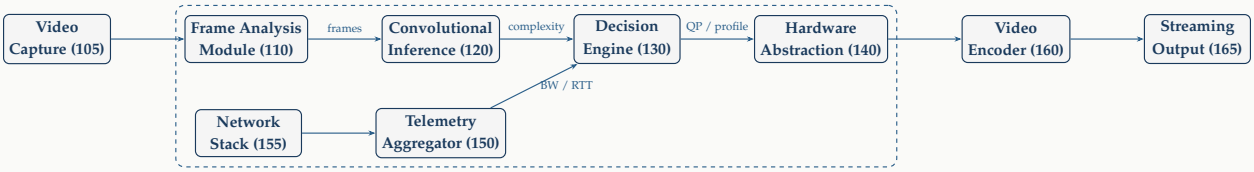

FIG. 1. Block diagram of the Adaptive Neural Compression Engine (ANCE) system. Reference numerals identify subsystems as defined in the Detailed Description. Dashed boundary delineates the inventive ANCE Core **(100)**. Arrows indicate primary data-flow direction.

Claims

What is claimed is:

1. A computer-implemented method for adaptive video compression, the method comprising:

- receiving, by a frame analysis module executing on at least one processor, a sequence of video frames and associated motion vector data from a video capture pipeline;
- generating, by a convolutional neural network having a parameter count not exceeding 2.5 million, a complexity score vector for each macroblock partition of each said video frame, wherein said complexity score vector has a dimensionality of at least 32 floating-point values;
- receiving, by a telemetry aggregator, a plurality of network bandwidth measurements and round-trip latency measurements;
- computing, by the telemetry aggregator, a filtered bandwidth estimate using a Kalman filter applied to said network bandwidth measurements;
- determining, by a decision engine, a per-macroblock quantization parameter and an encoding profile index based on said complexity score vector and said filtered bandwidth estimate; and
- encoding, by a video encoder, said video frames using said per-macroblock quantization parameters and said encoding profile index within a maximum end-to-end latency of 4 milliseconds per frame on a 1-gigahertz single-core processor.

2. The method of claim 1, wherein the convolutional neural network comprises depthwise-separable convolutional layers and at least one squeeze-and-excitation attention block.

3. The method of claim 1, wherein computing the filtered bandwidth estimate further comprises maintaining a circular buffer of at least 512 timestamped bandwidth samples and updating a Kalman filter state at each new measurement.

4. The method of claim 1, wherein the decision engine comprises a multi-layer perceptron having at least three fully connected layers with sigmoid-linear activation functions.

5. The method of claim 1, wherein the encoding profile index is selected from a set comprising at least a baseline profile, a main profile, and a high profile of the H.264/AVC video coding standard.

6. The method of claim 1, further comprising:

- benchmarking, at initialization, at least two hardware execution backends comprising a central processing unit backend, a graphics processing unit backend, and a neural processing unit backend; and
- selecting, for subsequent frame processing, the hardware execution backend whose mean per-frame inference latency satisfies the 4-millisecond latency budget.

7. The method of claim 6, further comprising, upon determining that no hardware execution backend satisfies the 4-millisecond latency budget, computing a fallback quantization parameter according to the formula:

$$q_{\mathrm{fb}} = \mathrm{clamp}\left(51 - \left\lfloor 43 \cdot \hat{b}_t / b_{\max} \right\rfloor,\ 18,\ 42\right)$$

8. A system for adaptive video compression, the system comprising:

- at least one processor;
- a memory storing instructions that, when executed by the at least one processor, implement:
  - a frame analysis module configured to receive video frames and compute per-macroblock motion vectors;
  - a convolutional inference module configured to generate complexity score vectors for each macroblock;
  - a telemetry aggregator configured to maintain Kalman-filtered bandwidth and latency estimates; and
  - a decision engine configured to jointly process said complexity score vectors and said estimates to produce per-macroblock encoding parameters.

9. The system of claim 8, further comprising a hardware abstraction layer configured to dispatch inference workloads to a selected one of a central processing unit, a graphics processing unit, and a neural processing unit based on runtime latency benchmarks.

10. The system of claim 8, wherein the convolutional inference module stores model weights in IEEE 754 half-precision floating-point format.

11. The system of claim 8, wherein the system is embodied as a dedicated integrated circuit module.

12. A non-transitory computer-readable medium storing instructions that, when executed by at least one processor, cause the processor to perform:

- analyzing a video frame using a depthwise-separable convolutional neural network to produce per-macroblock complexity scores;
- estimating available network bandwidth via a Kalman filter applied to a plurality of network telemetry measurements;
- selecting per-macroblock quantization parameters and an encoding profile using a trained multi-layer perceptron that jointly processes said complexity scores and said estimated bandwidth; and
- dispatching encoding parameters to a video codec within 4 milliseconds of receiving the corresponding video frame.

13. The computer-readable medium of claim 12, wherein analyzing the video frame further comprises extracting motion vectors from a hierarchical block-matching search having a maximum search range of ±32 pixels.

14. The computer-readable medium of claim 12, wherein the convolutional neural network is quantized to 8-bit integer precision via post-training quantization using a representative calibration dataset of at least 5,000 video frames.

Abstract

Abstract of the Disclosure

An Adaptive Neural Compression Engine (ANCE) is disclosed for real-time adaptive bitrate video encoding over bandwidth-constrained networks. The system integrates (1) a lightweight depthwise-separable convolutional neural network (fewer than 2.5 million parameters) that produces per-macroblock scene-complexity scores at frame rate; (2) a Kalman-filtered telemetry aggregator that maintains continuous estimates of network bandwidth and round-trip latency; and (3) a multi-layer-perceptron decision engine that jointly processes complexity and telemetry inputs to emit per-macroblock H.264/AVC quantization parameters and encoding profiles within a 4-millisecond latency budget on commodity hardware. A hardware abstraction layer provides transparent dispatch to CPU, GPU, or NPU backends. Experimental evaluation on the UVG-HD benchmark demonstrates a 1.4-point improvement in Mean Opinion Score (ITU-T P.910) versus rule-based adaptive bitrate baselines at equivalent mean throughput. The invention is applicable to live video streaming, video conferencing, and real-time surveillance transcoding on devices ranging from mobile handsets to cloud-deployed encoding clusters.

Attorney of Record	Michael T. Hargreaves, Reg. No. 62,441
Firm	Meridian IP Group LLP, 800 Capitol Mall, Suite 2400, Sacramento, CA 95814
Date of Signing	March 14, 2024

Signature of Attorney/Agent

Date (MM/DD/YYYY)

Printed Name

USPTO Registration Number