Encoder-Decoder Backbones for Variable-Aspect-Ratio Log Images

Raster well-log digitisation has to cope with images whose width varies by a factor of four, from a 3,200 by 480 floor to a 12,800 by 640 ceiling, with no fixed aspect ratio in between. The conventional response is to make the input regular before the network sees it, by resizing or by tiling into fixed crops, and then to spend modelling effort on the loss function. We argue the priority is inverted. CurveNet, the encoder-decoder backbone we built for this task, accepts the full variable-aspect-ratio range natively because its shape, not its objective, is built to absorb size: five stride-2 encoder stages divide the spatial dimensions away while a 128-dimensional bottleneck holds constant, two transformer attention layers operate on that fixed-width bottleneck regardless of input length, a GroupNorm grouping floored at sixteen keeps normalisation statistics stable when the batch is small, and five upsample decoder stages reconstruct a mask at the original resolution. The same architecture trains a sixteen-image multiclass batch through a custom collate path and a single-image binary batch under a tighter memory budget without any structural change. This document makes the case that this shape is the load-bearing decision. It walks the down-up cadence with a draggable stress bench that stretches the input across the full range, examines the attention bottleneck that gives the network a global receptive field over a long thin image, and traces how the variable-dimension batch is fed without tiling. It situates the design against the public encoder-decoder, normalisation, and attention literature it builds on, reports the figures the architecture made reachable, and states the engineering rule we now follow: choose the backbone for the input distribution first, and treat the loss as a tuning knob on top of a shape that already fits.

Tannistha Maiti
Senior AI Researcher

Narendra Patwardhan
Research Collaborator

Tarry Singh
Founder & CEO

VEERNET / CURVENET / ENCODER-DECODER / U-NET / TRANSFORMER-ATTENTION / GROUPNORM

CONTENTS

> The hardest thing about a scanned well log is not the noise on the curve. It is that the next log is a different size, and the one after that is different again. Solve the size problem in the architecture and the rest of the model gets simpler.

I

THE DECISION

Where this kind of model is actually won

Executive summary

There is a default order of operations in segmentation work. You take the architecture as a near-given, usually some flavour of encoder-decoder, you make the inputs regular so they fit it, and then you pour your effort into the loss function, because that is where the published gains seem to come from. We followed the opposite order when we built CurveNet, the backbone behind VeerNet, our system for digitising raster well logs, and this whitepaper is the argument for why.

The input to a raster-log digitiser is not regular and cannot be made regular cheaply. A scanned log is very wide and short, and its width varies across the corpus from a floor of **3,200** pixels to a ceiling of **12,800**, a factor of four, with the height moving from **480** to **640** and no fixed aspect ratio anywhere in between. The reflex answer is to tile: slice each image into fixed crops, run the network on every crop, and stitch the predicted masks back together. We did not tile. We built a backbone whose shape absorbs the variation directly, so a single checkpoint ingests the whole range natively.

The shape is an encoder-decoder with five stride-2 stages on the way down and five upsample stages on the way back, a **128**-dimensional bottleneck at the bottom carrying **two** transformer attention layers, and GroupNorm with its group count floored at **sixteen**. Each of those choices answers a specific consequence of variable input size. The five stride-2 stages divide the spatial dimensions away, so by the bottleneck the width has been reduced by a factor of thirty-two and the absolute size of the input has stopped mattering to everything downstream. The attention layers sit on that fixed-depth bottleneck and give the network a global receptive field over a long thin image at bounded cost. The GroupNorm floor keeps normalisation statistics stable when the batch is forced down to a

single image, which the binary task requires. The result is one architecture that trains a **sixteen**-image multiclass batch through a custom collate path and a **single**-image binary batch under a tighter memory budget, with no structural change between them.

Our claim is narrow and falsifiable: for this input distribution the backbone is the load-bearing decision, and the loss function is a tuning knob applied on top of a shape that already fits. The rest of this document defends that claim, instrument by instrument.

THE RANGE ONE BACKBONE HAS TO SPAN

3,200
Minimum synthetic log width, pixels

12,800
4x range
Maximum synthetic log width, pixels

128
Bottleneck embedding dimension, fixed across the range

16 vs 1
same shape
Batch size, multiclass versus binary

II

WHY NOW

The size problem the corpus hands you

Variable aspect ratio is the structural property, not an edge case

Most public segmentation benchmarks are built from images that share a resolution, or that have been cropped to one before anyone trains on them. Raster well logs do not give you that. A log is a tall narrow strip in the physical world but a scanned image of it is stored wide, and the scan dimensions depend on the document, the scanner, the depth interval, and the era. Across the synthetic corpus we generate to train on, and across the real Texas Railroad Commission scans the system has to serve, the width ranges over a factor of four and the height moves too. There is no single resolution to standardise on without throwing information away.

This is not a corner case to be patched late. It is the first-order property of the data, and it forces an early decision that quietly shapes everything after it. Either you make the input regular before the network, or you make the network tolerate irregular input. The classical-vision lineage of this task, gridline elimination and morphological extraction, sidestepped the question because it never batched images in the first place [5]. A deep-learning system does batch, and the moment you batch you confront the size problem head on.

The tiling reflex and what it costs

The tiling answer is attractive because it is simple to reason about. Fix a crop size, slice every image into crops of that size, run the model on a stack of identical-shaped crops, and reassemble. Your tensors are regular, your batching is trivial, and you can reuse any off-the-shelf fixed-input architecture unchanged.

The cost shows up at the seams. A curve trace that crosses a crop boundary is seen by two crops, each with only half the context, and the model has to make a consistent decision twice and hope the stitch agrees. For a target that is frequently a single pixel wide and runs the entire length of the image, those seams are not rare events at the margins; they are everywhere, because the thing you are segmenting is precisely the long continuous structure that tiling chops up. Every boundary is an opportunity for a discontinuity that a human would never introduce. You can mitigate it with overlapping crops and blending, but now you are running the model several times per pixel and writing stitch logic that is itself a source of bugs.

Native ingest moves the cost into the architecture, once

Accepting the image whole moves the entire problem into one place, the shape of the network, and pays for it once at design time rather than repeatedly at inference. There are no seams to stitch because there are no crops. The long continuous curve is seen in one pass with full context end to end. The price is that the architecture now has to be genuinely size-tolerant, and that is a real engineering constraint rather than a free lunch. The remainder of this whitepaper is about how the encoder-decoder shape pays that price, and why paying it in the architecture is the better trade for this product than paying it in a tiling pipeline.

> **THE TRADE WE MADE EXPLICIT**
>
> Tiling spends a fixed-input model and an unbounded amount of stitch complexity at inference time. Native ingest spends a one-time architectural constraint at design time. For a target that is a long continuous thin curve, the seams tiling introduces are the worst possible failure mode, so we paid the architecture cost instead.

III

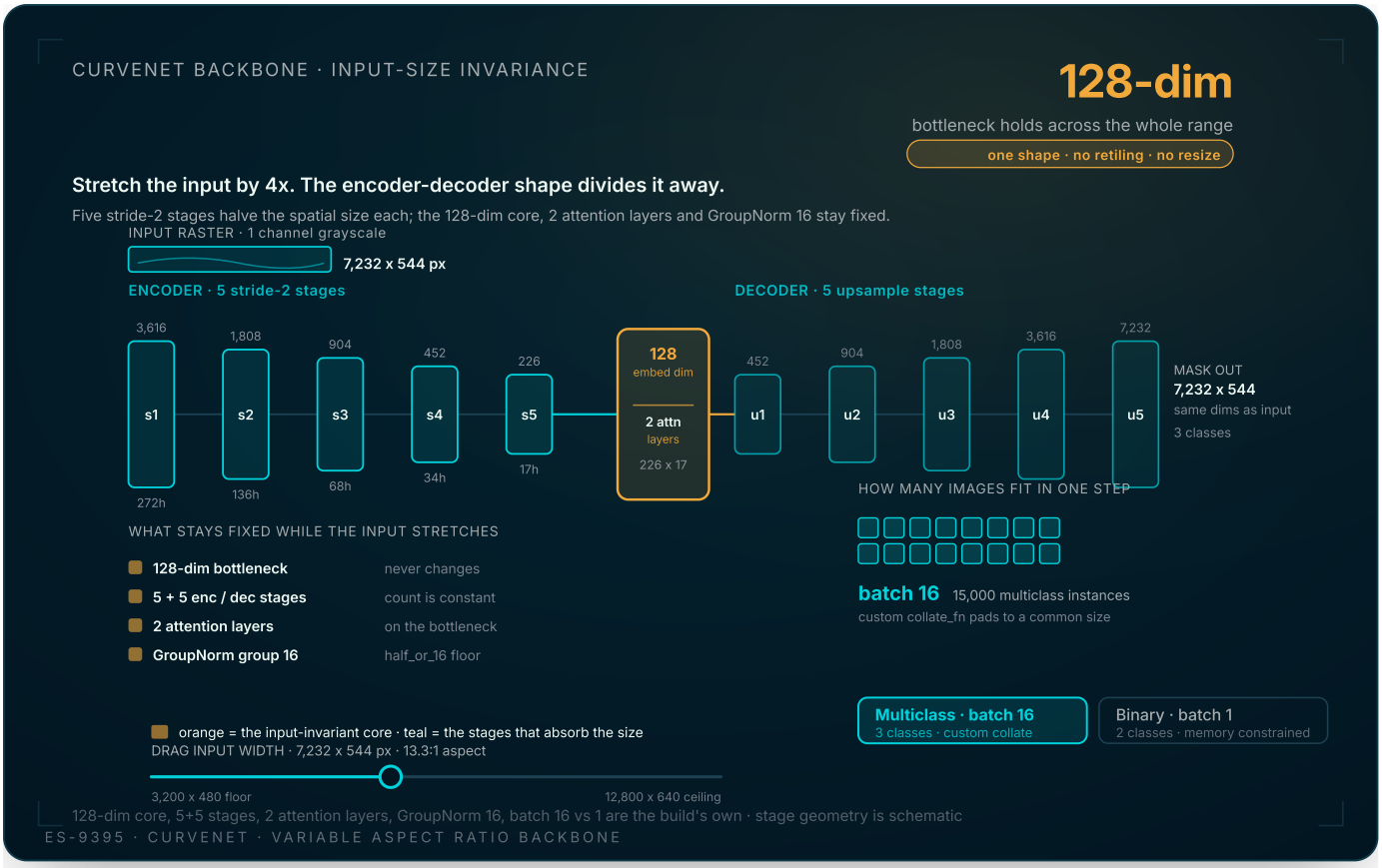
THE BACKBONE

A shape that divides the input size away

How CurveNet ingests the whole range

CurveNet is an encoder-decoder, the same family as U-Net [1], specialised for an image that is long, thin, and of unpredictable width. The contracting path has five stages, each a stride-2 step that halves the spatial dimensions while the feature depth grows. Five halvings reduce a side length by a factor of thirty-two, so a **12,800**-pixel-wide input arrives at the bottleneck at a four-hundred-wide feature map, and a **3,200**-pixel-wide input arrives at a hundred-wide one. The bottleneck depth is the same **128** dimensions either way. This is the crux: after five stride-2 stages the absolute input size has been quotiented out, and everything from the bottleneck onward operates on a representation whose depth is fixed and whose spatial extent is small enough not to matter.

The residual blocks that make those five deep stages trainable without degradation are the standard residual-learning construction [4]; we did not reinvent them, we relied on them so the encoder could go deep enough to do the dividing. The decoder mirrors the encoder with five upsample stages that restore the spatial resolution back to the input size and emit a per-pixel mask, so the output is the same dimensions as whatever came in. Drag the stress bench below from the floor to the ceiling and watch the cadence: the input plate stretches, the per-stage spatial dimensions fall by powers of two, the orange core in the middle holds at 128 dimensions, and the output comes back at full resolution.



The lead exhibit for the architecture argument. A single CurveNet encoder-decoder shape spans the whole synthetic-log range, from the 3,200 x 480 floor to the 12,800 x 640 ceiling, with no retiling and no resize. Drag the input width and watch the funnel work: the five stride-2 encoder stages halve the spatial size at each step, the five upsample decoder stages put it back, and the orange core in the middle never moves. That core is the 128-dimensional bottleneck with two transformer attention layers and a GroupNorm group floored at 16. It is the thing the whitepaper claims is what makes input-size invariance possible. The right panel shows the other half of the story: the multiclass path packs 16 images into a step through a custom collate that pads to a common size, while the binary path is memory bound by raw width and runs one image at a time. The fixed figures (128-dim bottleneck, 5 plus 5 stages, 2 attention layers, GroupNorm 16, batch 16 versus 1) are the build's own; the drawn stage geometry is schematic, showing the spatial-halving cadence rather than the exact channel tally at each layer.

What the bench is meant to make obvious is that the only thing in the diagram that still scales with the input is the spatial footprint of the bottleneck feature map, and that footprint is small. Everything that carries the model's capacity, the channel depths, the bottleneck width, the attention layers, the normalisation groups, is invariant to how wide the input was. That invariance is the whole point. A model whose parameters depend on input size needs a model per size, which is the tiling world in a different costume. A model whose parameters are size-independent needs one checkpoint for the range, which is what we ship.

The 128-dimensional bottleneck is a deliberate width, not a default

It would be possible to read the 128-dimensional bottleneck as an arbitrary hyperparameter. It is not. It is the width at which the attention layers that sit on the bottleneck stay affordable while still being expressive enough to carry the global structure of a multi-curve track. Too narrow and the attention has nothing to work with; too wide and the attention cost grows without buying recovery on a target this sparse. The bottleneck width is the meeting point of two pressures, the depth the representation needs and the cost the attention can bear, and 128 is where those balanced for this task.

GroupNorm at sixteen is what lets one shape serve two batch sizes

Normalisation is where a lot of size-tolerant designs quietly break. BatchNorm computes its statistics across the batch, so when the batch is a single image, which the binary task forces because of memory, the statistics are computed over one example and are unstable. We use GroupNorm instead [2], with the group count floored at sixteen, the rule we call half_or_16. GroupNorm computes its statistics over groups of channels within a single example, so it does not care how many images are in the batch. That is precisely the property that lets the identical backbone train a **sixteen**-image multiclass batch and a **single**-image binary batch without changing anything about its normalisation behaviour. The batch size is dictated by memory and by the task; the normalisation is decoupled from it by construction.

Native variable-aspect ingest

- Accepts the full 3,200 to 12,800 pixel width range with no resize and no tiling
- The five stride-2 stages divide the spatial size away; the bottleneck width is fixed
- One checkpoint serves the whole range rather than a model per crop size
- Removes the stitching seams that fixed-crop tiling introduces

Constant-width attention bottleneck

- Two transformer attention layers operate on the 128-dimensional bottleneck
- Self-attention gives a global receptive field over a long thin image
- Links a feature in one part of the track to its continuation far away
- Cost stays bounded because attention runs at the downsampled bottleneck, not the raster

Small-batch-stable normalisation

- GroupNorm with a group count floored at sixteen, the half_or_16 rule
- Statistics do not collapse when the binary batch is a single image
- Lets the same shape train at batch 16 multiclass and batch 1 binary
- Decouples normalisation quality from the batch size the input forces

IV

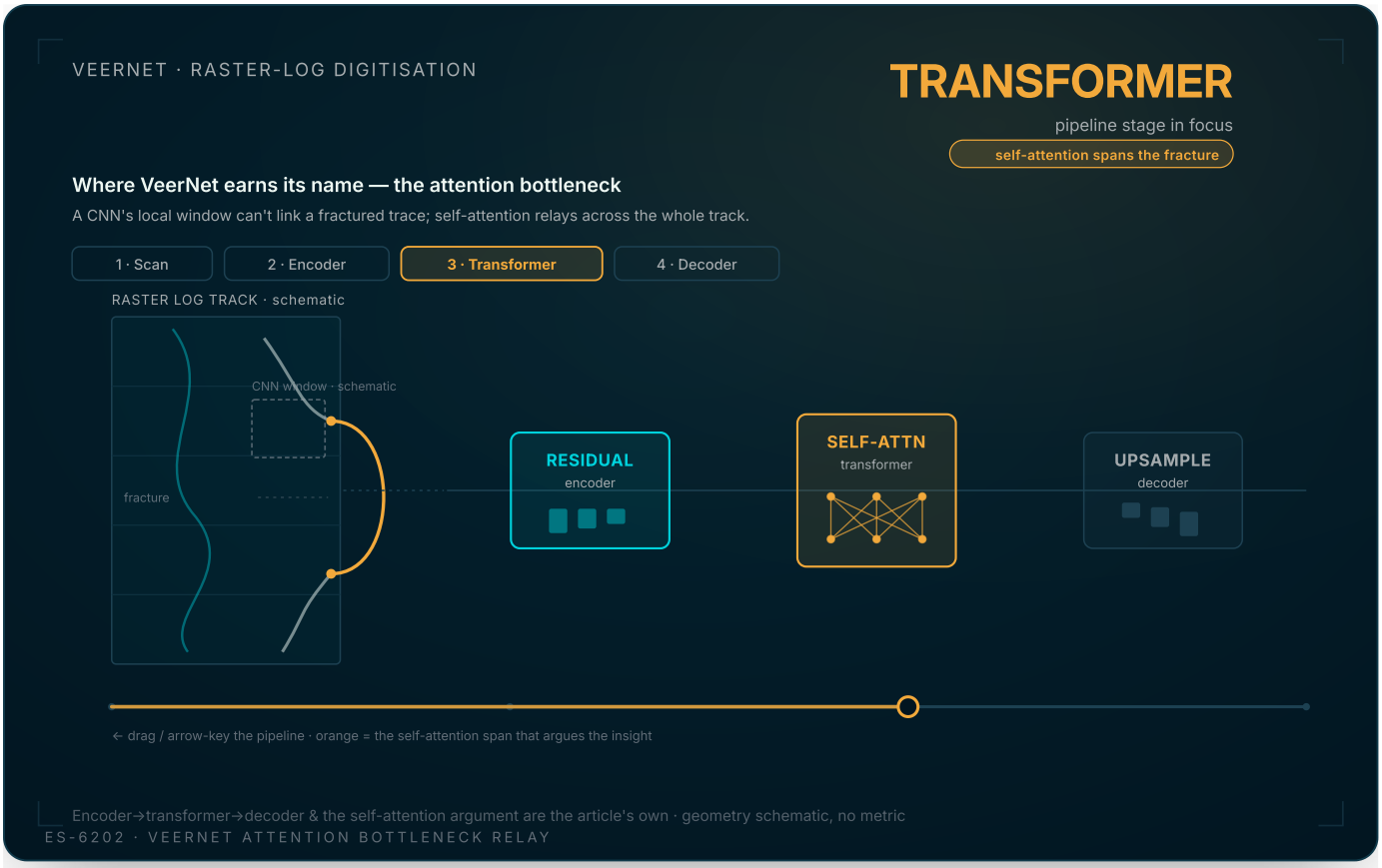
THE BOTTLENECK

A global receptive field over a long thin image

Why attention belongs on the bottleneck, and only there

A pure convolutional encoder-decoder has a local receptive field. Each output position is informed by a neighbourhood of the input, and although stacking stride-2 stages widens that neighbourhood, a convolutional network still reasons mostly locally. For most images that is fine. For a long thin log it is a real limitation, because the things you need to relate are far apart along the length of the image. A feature in one part of a track and its continuation a long way down the same track are part of one structure, and a network that only ever sees a local window cannot link them.

This is the gap the two transformer attention layers on the bottleneck close. Self-attention lets every position attend to every other position [3], which is a global receptive field by construction. We place the attention at the bottleneck rather than on the raster for a hard practical reason: attention cost grows with the number of positions, and the bottleneck has the fewest positions in the whole network because five stride-2 stages have already reduced the spatial extent by a factor of thirty-two. Running attention on the full-resolution raster of a 12,800-wide image would be ruinous. Running it on the downsampled bottleneck is affordable, and it is the one place in the network where a global view is both cheap and useful, because the bottleneck is where the representation is most abstract and most spatially compressed. The companion exhibit traces this relay: a structure in one part of the track and its continuation elsewhere, which a local convolution cannot connect, get bridged by the attention span at the bottleneck.



VeerNet's mechanism, not its metrics. A scanned multi-curve log track flows through a residual encoder, a self-attention transformer bottleneck, and an upsampling decoder that emits a per-pixel curve mask. A CNN's local receptive field (grey window) cannot link a fracture in the upper third of a track to the trace continuing through the lower third; the orange self-attention span bridges exactly that gap — which is, the article says, where VeerNet earns its name. Drag the scrubber (or arrow-key it) to advance the pipeline and watch the long-range attention relay stitch the two disconnected fragments. Every shape is structural/schematic: the article describes the architecture and the self-attention argument in prose but gives no kernel sizes or span lengths, so the instrument carries no benchmark numbers.

The reason this matters for the aspect-ratio argument is subtle but central. The wider the input, the further apart the related structures can be, so the longer the range over which the network must reason. A local convolutional field that was adequate on a 3,200-wide image is proportionally less adequate on a 12,800-wide one. Attention does not degrade that way. Its receptive field is global regardless of how long the bottleneck sequence is, so the network's ability to relate distant structures does not weaken as the input grows. The attention bottleneck is therefore not just a quality improvement; it is part of what makes the single backbone hold up across the full width range rather than working well at the floor and poorly at the ceiling.

> Two layers of attention on the bottleneck cost almost nothing because five stride-2 stages already shrank the sequence. That is the entire reason the global view is affordable on a 12,800-pixel image.

V

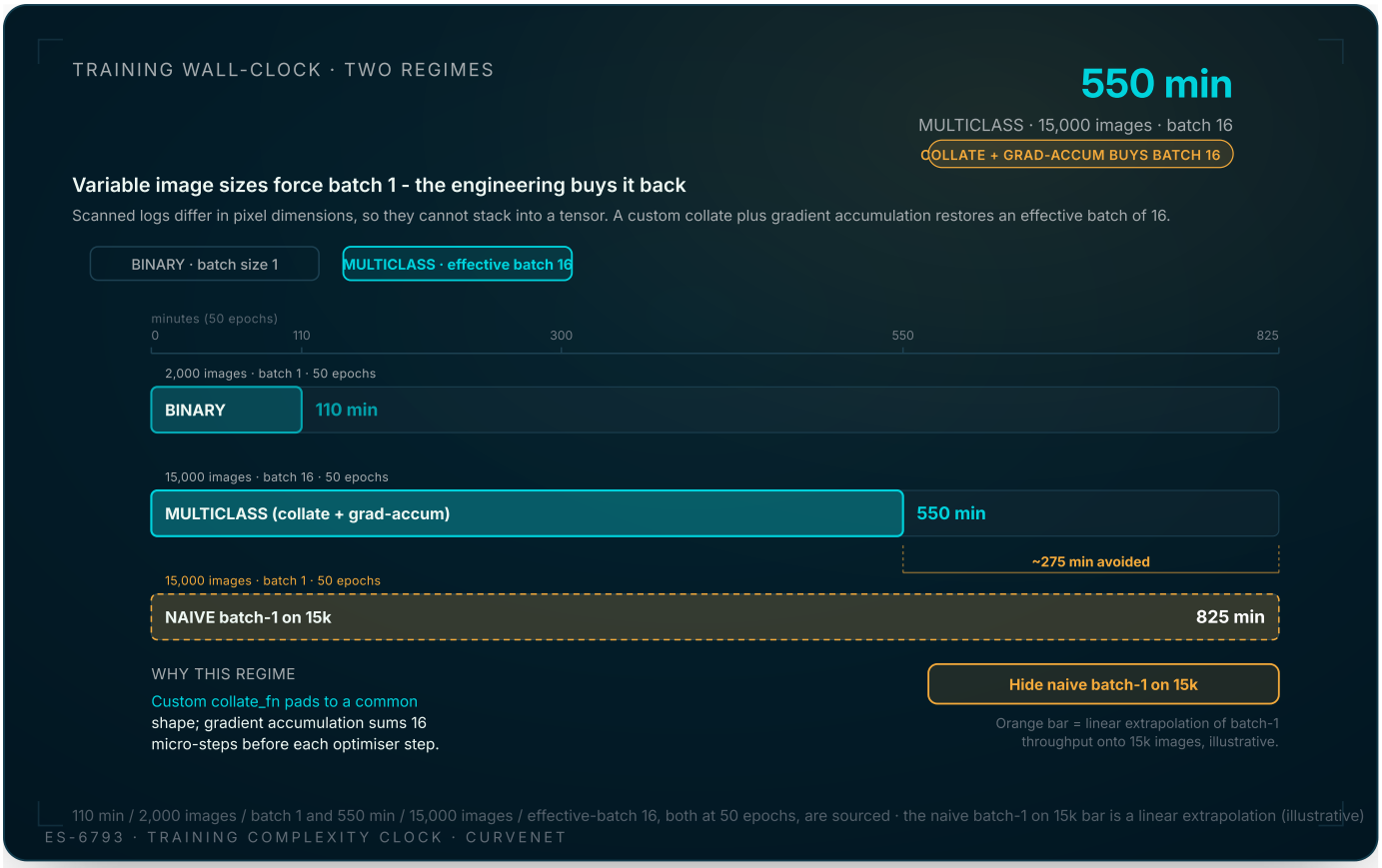
THE DATA PATH

Feeding a batch of images that are all different sizes

The collate function is part of the architecture decision

Choosing to ingest variable-size images natively creates a problem the architecture diagram does not show: you cannot stack images of different sizes into a single tensor, and a deep-learning training step wants a tensor. The naive resolution is batch size one, one image per step, which the binary task accepts because its memory budget is tight anyway. For the multiclass task, batch size one would leave the accelerator starved and the wall clock long, so we feed an effective batch of **sixteen** through a custom collate path that assembles a batch from differently sized examples rather than forcing every image to a common crop. This is the data-loader counterpart to the size-tolerant backbone: the network can accept any size, and the collate makes a batch of mixed sizes feedable.

The exhibit below makes the consequence concrete. The binary regime trains **2,000** instances at batch one and finishes fifty epochs in **110** minutes; the multiclass regime trains **15,000** instances at the effective batch of sixteen and finishes fifty epochs in **550** minutes. Toggle the naive batch-one extrapolation on the larger dataset to see the wall clock the collate path avoided.



Scanned well-log images arrive at wildly different pixel dimensions, so they cannot be stacked into a tensor; the naive fix is batch size 1, which under-feeds the GPU and stretches the wall-clock. CurveNet's answer is a custom collate_fn plus gradient accumulation, which buys back an effective batch of 16 without the memory blowup. Pick a regime; toggle the naive batch-1 on 15k bar to see the wall-clock the fix avoided. The 110 min / 2,000 images / batch 1 and 550 min / 15,000 images / effective-batch 16 figures (both at 50 epochs) are the handover's own; the orange naive bar is a linear extrapolation of the batch-1 throughput onto 15k images and is flagged as illustrative.

The point this makes for the backbone argument is that the two regimes share one architecture. There is no multiclass model and binary model with different shapes. There is one CurveNet, and the difference between the two runs lives entirely in the data path and the batch size, both of which the size-tolerant shape and the GroupNorm choice were designed to accommodate. The binary task at batch one and the multiclass task at batch sixteen are the same network meeting two different memory budgets, and the architecture absorbs the difference without flinching because that absorption was the design goal.

What the batch difference does and does not change

It is worth being precise about what the batch-size split is and is not. It is a memory and throughput decision, not an architectural one. The single-image binary batch is constrained by the raw width of the image; a 12,800-pixel-wide image at full resolution is a large activation map, and at batch one the memory budget is already committed. The sixteen-

image multiclass batch is feasible because the multiclass synthetic corpus is generated within dimensions the collate can pad to a common working size economically. In both cases the normalisation is GroupNorm at the same group floor, the bottleneck is the same 128 dimensions, the attention is the same two layers, and the stage count is the same five down and five up. The batch size is the one thing that varies, and it varies in the data path, which is exactly where you want a memory decision to live, rather than in the model.

> **“One shape, two batch sizes, two tasks. The only thing that changes between the binary and multiclass runs is how many images we can afford to feed at once, and that is a data-loader decision, not a model decision.”**

— FROM OUR TRAINING CONFIGURATION NOTES

VI

THE EVIDENCE

What the shape made reachable

The figures the architecture put within range

A whitepaper about architecture has to connect the shape to the numbers, carefully, because the numbers come from the whole system and not from the backbone alone. We are explicit about that. The headline reconstruction figures the system reached, a peak coefficient of determination of **0.9891** against native LAS data, a lowest mean absolute error of **0.0132**, and a lowest mean squared error of **0.0004**, are properties of the trained model on its deliverable, and the choice of loss function moved them around within the ablation. What the architecture contributed is the precondition: those numbers were reached on the full variable-aspect-ratio corpus, ingested natively, without tiling and without a model per crop size. A shape that could not span the range would have made the loss ablation moot, because there would have been no single model to ablate.

REACHED ON THE NATIVELY-INGESTED CORPUS

0.9891
Peak R-squared on the reconstructed curve, against LAS

0.0132
Lowest mean absolute error on the curve

0.0004
Lowest mean squared error on the curve

1
no per-crop models
Number of checkpoints serving the whole width range

The training economics tell the same story from the cost side. The multiclass run over **15,000** instances completed fifty epochs in **550** minutes, and the binary run over **2,000** instances completed fifty epochs in **110** minutes, on the same backbone. Those are not the numbers of a system fighting its own input format. They are the numbers of a shape that fits the data, where the wall clock is spent on learning rather than on working around size. The five-loss ablation that sits on top of this backbone, which the companion evaluation whitepaper covers in full, is a comparison of objectives against a fixed architecture; it could only be run because the architecture held the range constant underneath it.

Reading the architecture as the controlled variable

The cleanest way to see the backbone's contribution is to notice what was held fixed while the losses varied. Across the five-loss ablation the network shape did not change. The same five stride-2 stages, the same 128-dimensional attention bottleneck, the same GroupNorm floor, the same five upsample stages carried every loss function. The

differences in the resulting curves came from the objective, not the shape, which is the definition of a controlled experiment with the architecture as the control. That the ablation was possible at all, that every loss ran on identical inputs spanning the full width range, is the architecture doing its job silently. The loss ablation is the visible science; the backbone is the bench it was run on.

VII

THE BUILD

The down-up cadence in detail

Methods deep-dive

This section sets out the shape precisely enough to reproduce the reasoning, while flagging where a figure is a fixed engagement number and where a drawn quantity in the exhibits is schematic.

The encoder is five stride-2 stages built from residual blocks [4]. Each stage halves the spatial dimensions, so the cumulative reduction across the five stages is a factor of thirty-two in each spatial dimension. The input is a single-channel grayscale raster, because a scanned log carries its signal in intensity rather than colour. The bottleneck is **128**-dimensional, and on it sit **two** transformer attention layers [3] that provide the global receptive field discussed above. The decoder is five upsample stages that mirror the encoder and restore the output to the input resolution, producing a per-pixel mask: two foreground classes plus background in the multiclass setting, and a binary foreground-against-background mask in the binary setting.

Normalisation is GroupNorm [2] with the group count floored at sixteen, the half_or_16 rule, chosen so the statistics are stable at the single-image batch the binary task needs. Batching is the split already described: an effective batch of **sixteen** for the multiclass task via the custom collate path, and a batch of **one** for the binary task under its tighter memory budget. Training is fifty epochs in both regimes, **550** minutes for the **15,000**-instance multiclass run and **110** minutes for the **2,000**-instance binary run.

Two honesty notes about the exhibits. First, in the stress bench the per-stage block heights and the count of glyphs drawn are schematic; they communicate the spatial-halving cadence and the topology, not the exact channel tally at every layer. Second, in the training clock the naive batch-one extrapolation on the larger dataset is a linear projection of the batch-one throughput, flagged on the canvas as illustrative rather than measured. The fixed

figures, the **3,200** to **12,800** width range, the **480** to **640** height range, the **128**-dimensional bottleneck, the **five** plus **five** stages, the **two** attention layers, the batch of **sixteen** versus **one**, the GroupNorm floor of **sixteen**, and the training times, are the engagement's own.

> ### WHAT IS FIXED AND WHAT IS SCHEMATIC
>
> Every numeric claim in this whitepaper is a figure from the engagement archive. The geometry drawn inside the exhibits, block sizes and glyph counts, is schematic and exists to show topology and cadence. Where an exhibit shows a projected rather than a measured value, it says so on the canvas.

VIII

THE POSITION

What this argues for the next model

Implications and roadmap

The operating rule we take from this build is an ordering rule. When the input distribution has a strong structural property, and variable aspect ratio over a four-fold range is a strong structural property, the architecture should be chosen to fit that property first, and the loss function should be treated as a tuning knob applied afterward. The common order, fix the architecture, regularise the input, tune the loss, inverts the priority and pushes a first-order problem into a preprocessing step where it becomes tiling seams and per-crop models. We think that order is wrong for any task where the input shape is the dominant difficulty.

For the roadmap, this has three consequences. First, future curve types and future log formats should be absorbable by the same backbone as long as their dimensions stay within the range the shape was built to divide away, which means new data can often be added without a new architecture. Second, the attention bottleneck is the natural place to invest if longer or more complex tracks are added, because it is where global structure is reasoned about and where the cost of a wider input is already controlled. Third, the data path, the collate function and the batch strategy, is a first-class part of the design and should evolve alongside the model rather than being treated as plumbing, because it is what makes the size-tolerant shape feedable in practice.

There is a broader point about where the value of these systems sits. The upstream-AI literature keeps arriving at the conclusion that a model's worth is set by the workflow it serves rather than by an offline score in isolation [6]. A digitiser that needs a different model for every scan size, or that introduces stitch artefacts at every crop boundary, is harder to operate and trust than one checkpoint that takes any log and returns a clean mask. The architecture choice we are defending here is, in the end, an operability choice as

much as a modelling one. The backbone that fits the input distribution is the backbone that an interpreter can rely on without thinking about pixel dimensions, and that reliability is the thing the engagement was actually buying.

THE LOAD-BEARING POINTS

1. Variable aspect ratio over a **4x** range, from **3,200** to **12,800** pixels wide, is the first-order property of the data, not an edge case. The architecture should answer it first.
2. Five stride-2 encoder stages divide the spatial size away so the **128**-dimensional bottleneck and everything downstream are invariant to input width. One checkpoint serves the whole range, with no tiling and no per-crop models.
3. Two attention layers belong on the bottleneck and only there: it has the fewest positions after five halvings, so a global receptive field over a long image is affordable exactly where it is most useful.
4. GroupNorm floored at **sixteen** keeps normalisation stable at the single-image batch the binary task forces, which is what lets one shape train at batch **16** multiclass and batch **1** binary unchanged.
5. Choose the backbone for the input distribution, then treat the loss as a knob on top. The shape is the load-bearing decision; the objective is tuned against a shape that already fits.

GLOSSARY

Aspect ratio
The ratio of width to height. Well-log scans run very wide and short, and the ratio is not fixed, which is the property the backbone has to tolerate without retiling.

Bottleneck
The narrowest point of the encoder-decoder, where spatial resolution is lowest and feature depth is highest. Here it is 128-dimensional and carries the two attention layers.

Custom collate
A data-loader function that assembles a batch from examples of different sizes. It is what lets a sixteen-image multiclass batch form despite each log image having different pixel dimensions.

Encoder-decoder
A network shape with a contracting path that reduces spatial resolution while building feature depth, then an expanding path that restores resolution to produce a dense output. For segmentation the output is a per-pixel mask at the input size.

GroupNorm
Group Normalization. A normalisation that computes statistics over groups of channels within a single example, so it does not depend on batch size. We floor the group count at sixteen.

Self-attention
A mechanism that lets every position in a feature map attend to every other position, giving a global receptive field. Placed on the bottleneck so the network can relate distant parts of a long log image.

Stride-2 stage
A convolution step that halves the spatial dimensions. Five of them in sequence reduce a side length by a factor of thirty-two, which is how a 12,800-wide image becomes a small bottleneck feature map.

Tiling	Cutting a large image into fixed-size crops, running the model on each, and stitching the outputs. The common workaround for variable input size, and the one this backbone is built to avoid.

References

1. Ronneberger, O., Fischer, P., Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. MICCAI. https://arxiv.org/abs/1505.04597
2. Wu, Y., He, K. (2018). *Group Normalization*. ECCV. https://arxiv.org/abs/1803.08494
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I. (2017). *Attention Is All You Need*. NeurIPS. https://arxiv.org/abs/1706.03762
4. He, K., Zhang, X., Ren, S., Sun, J. (2016). *Deep Residual Learning for Image Recognition*. CVPR. https://arxiv.org/abs/1512.03385
5. Yuan, B., Yang, Q. (2019). *Digitization of Well-Logging Parameter Graphs Based on Gridlines-Elimination Approach*. Journal of Petroleum Exploration and Production Technology. https://link.springer.com/article/10.1007/s13202-019-0656-3
6. Koroteev, D., Tekic, Z. (2021). *Artificial intelligence in oil and gas upstream: Trends, challenges, and scenarios for the future*. Energy and AI. https://www.sciencedirect.com/science/article/pii/S2666546820300033

Get the full whitepaper

This page is the long-form summary. The complete whitepaper adds the full stage-by-stage tensor shapes across the width range, the collate-path implementation detail, the memory accounting behind the batch-16-versus-1 split, and the per-loss results measured on top of this fixed backbone.

Encoder-Decoder Backbones for Variable-Aspect-Ratio Log Images

Published June 2023. Generated 2026-07-22.
https://www.earthscan.io/whitepapers/encoder-decoder-backbones-variable-aspect-ratio-logs