LAS Service

A research model that wins on a validation set is not a product, and the distance between the two is mostly engineering and product design rather than more training. We built VeerNet, an encoder-decoder segmentation network with transformer attention refinement, as a bespoke deliverable for a US onshore operator: it digitised scanned raster well logs into spec-compliant LAS and CSV files, trained on roughly 15,000 synthetic curves, reaching a peak coefficient of determination of 0.9891, a lowest mean absolute error of 0.0132, and a 20x to 200x speedup over manual tracing. This whitepaper documents the productisation of that one-off capability into a repeatable, hosted, self-serve service, where a user uploads a scan, the model digitises it, a reviewer corrects the imperfect prediction, and a compliant file is downloaded. We treat productisation as three coupled engineering decisions. First, the shape change: a bespoke project is built once and billed once, while a service amortises one trained model across many scans, which inverts the unit economics and forces a different posture toward cost, isolation, and idempotency. Second, the flow: the four self-serve stages of upload, digitise, review, and download, where the review stage is not a convenience but the load-bearing element, because a peak curve-class intersection-over-union of only 0.51 means the model is excellent at most of the trace and wrong on a minority of it, and the product must let a human see and fix exactly that minority. Third, the throughput dividend: one build pointed at a reference archive of 136,771 scanned images returns tracing hours that scale with the speedup, the compounding payoff that justifies the productisation work. We embed three interactive instruments, a bespoke-versus-hosted ledger, the upload-to-download product flow, and a one-model-many-scans throughput meter, and close with the working rule we adopted, that the model is the easy half and the service is the half that decides whether anyone outside the original engagement ever benefits from it.

Tarry Singh
Founder & CEO

Tannistha Maiti
Senior AI Researcher

VEERNET / PRODUCTISATION / ML-SERVING / RASTER-LOG-DIGITISATION / SCAN-TO-LAS / HUMAN-IN-THE-LOOP

CONTENTS

> "The model was the part we could write a paper about. The service was the part that decided whether anyone other than one operator would ever use it."

I

THE TURN

From a deliverable we hand over to a service they drive

What we had, and what we did not have

We built VeerNet to solve a concrete problem for a US onshore operator. The operator held a large archive of well logs that existed only as scanned paper, raster images of curves drawn on grid paper, and those scans were useless to any modern petrophysics workflow until the curves inside them became numbers. The manual route was a person with a digitising tablet tracing each curve by hand, slow, expensive, and unrepeatable at the scale of an archive. We trained an encoder-decoder segmentation network with a transformer attention refinement on the bottleneck, on roughly **15,000** synthetic curves, to find those traces and convert them into depth-indexed values. It worked. On the hardest curve it reached a coefficient of determination of **0.9891**, its lowest mean absolute error was **0.0132**, and it digitised between **20x and 200x** faster than a person doing the same work by hand [4].

That is a research result, and it was delivered as a research result. The engagement was a bespoke deliverable: a fixed scope, a fixed timeline, a fixed price, and a handover of artefacts at the end. We ran it as **4** project blocks, on an accelerated **16-week** track staffed at **6** full-time engineers, against a standard **32-week** track at **4**, priced at **EUR 180,000** accelerated versus **EUR 100,000** standard. At the end of that, one operator had their logs digitised and we had a model in a repository. What we did not have was anything a second operator could use, or a way for the first operator to digitise the next batch of scans without commissioning us again.

The turn this whitepaper describes is the decision to close that gap. We stopped asking how do we digitise these logs and started asking how do we let anyone digitise any log, themselves, without us in the room. The answer is not a better model. The model was already good enough to be useful. The answer is a service: a hosted, self-serve flow where a user uploads a scan, the model digitises it, a reviewer fixes what the model got wrong,

and a spec-compliant LAS or CSV comes out the other end. Building that service is mostly engineering and product design, and very little additional training, which is the first and most important thing we learned, and the thing the machine-learning-systems literature has said for a decade: the model is a small box inside a large system, and the system is where the work and the cost live [1].

THE BESPOKE ENGAGEMENT, IN ITS OWN NUMBERS

0.9891

research result

Peak R-squared on the hardest curve

20x to 200x

Speedup over manual tracing

4 blocks

16-week track

Project structure

136,771

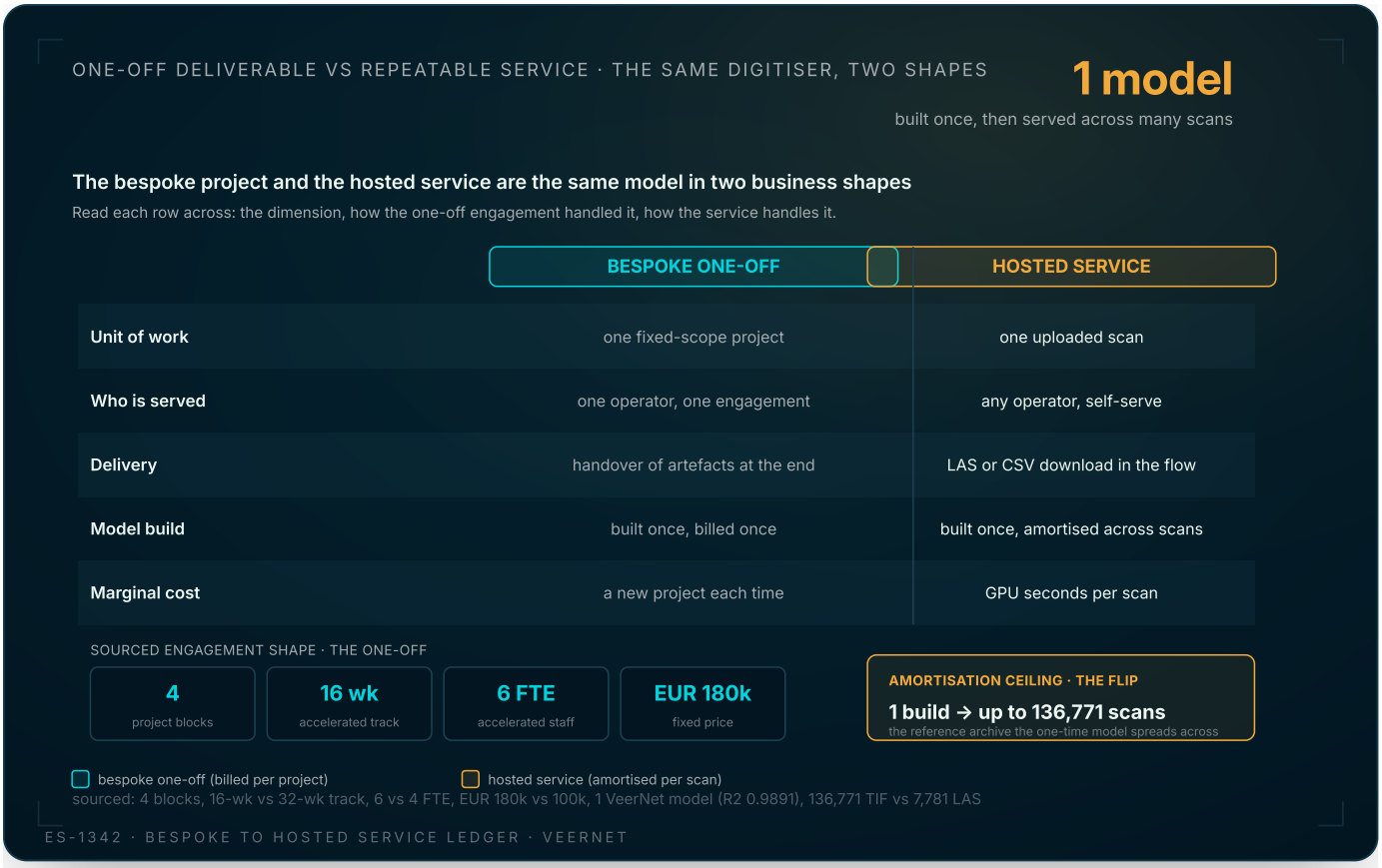
Scanned images in the reference archive

Why a one-off is not a service, even when the model is identical

It is tempting to think that once the model exists, turning it into a service is a matter of putting a web form in front of it. It is not, and the reasons are structural rather than cosmetic. A bespoke deliverable and a hosted service can run byte-for-byte the same model weights and still be entirely different products, because the thing that changes is not the model, it is everything around the model: who pays, how the output is delivered, what has to scale, and what happens when the input is wrong.

In the bespoke engagement, the unit of work was a project. We scoped a known set of scans, ran them, checked the results ourselves, and delivered. There was no concurrency problem, because there was one of us and a finite batch. There was no idempotency problem, because we ran each scan once under supervision. There was no untrusted-input problem, because we curated the inputs. There was no cost-per-request problem, because the cost was the project, billed once. Every one of those non-problems becomes a real problem the moment the same model is exposed as a service that strangers drive. A self-serve service has to assume concurrent users, repeated and retried requests, inputs it has never seen and cannot curate, and a cost that is now measured per scan rather than per project. The deployment-challenges literature is essentially a catalogue of these exact transitions and the ways they go wrong [3].

The first instrument lays this out as a ledger. It reads the same digitiser in two business shapes, the bespoke one-off on the left and the hosted service on the right, row by row across the dimensions that actually change. The point of the ledger is that the model column would be identical in both, so it is not on the ledger at all. What is on the ledger is the shape, and the orange figure, the single element that argues, is the amortisation: one model build spread across up to **136,771** scans rather than charged to one project. That is the property that makes a service worth building, and it is the property a one-off can never have.



What changes when a one-off deliverable becomes a repeatable service, read as a ledger. The left column is the engagement as we ran it: a fixed-scope project, staffed and billed once, handing one operator one set of digitised logs. The right column is the same digitiser packaged as a self-serve scan-to-LAS service, where the model is built once and amortised across every scan every operator uploads. The sourced facts are the engagement's own: 4 project blocks over a 16-week accelerated track against a 32-week standard track, 6 FTE against 4, a fixed accelerated price of EUR 180,000 against EUR 100,000 standard, one VeerNet model trained on 15,000 synthetic curves to a peak R-squared of 0.9891, and a reference archive of 136,771 TIF scans against 7,781 LAS files. The orange figure is the only one that argues: the amortisation multiplier, the single property that flips the unit economics once one trained model serves many scans instead of one project.

Reading the ledger

Walk the rows and the transition becomes concrete. The unit of work goes from one fixed-scope project to one uploaded scan, which means the entire billing, queuing, and accounting model has to be rebuilt around the scan as the atom. Who is served goes from one operator in one engagement to any operator, self-serve, which is the concurrency and isolation requirement stated plainly. Delivery goes from a handover of artefacts at the end of a project to a download in the flow, seconds after the upload, which forces the file generation to be automated and spec-correct rather than checked by hand. The model build goes from built once, billed once to built once, amortised across scans, which is the line that flips the unit economics.

The marginal-cost row is the one that surprises people. In the bespoke world, the marginal cost of the next operator was an entire new project, with its own scoping, staffing, and timeline. In the service world, the marginal cost of the next scan is GPU seconds. That is a difference of several orders of magnitude in what it costs to serve one more unit of demand, and it is the whole reason productisation pays. But it comes with a tax the bespoke model never paid: the service has to be running, and reliable, and safe to point a stranger at, all the time, whether or not anyone is using it. The amortisation card on the right of the instrument is the optimistic side of that ledger; the work described in the rest of this document is paying the tax that makes the optimistic side real.

II

THE FLOW

Upload, digitise, review, download

The product is four stages, and only one of them is the model

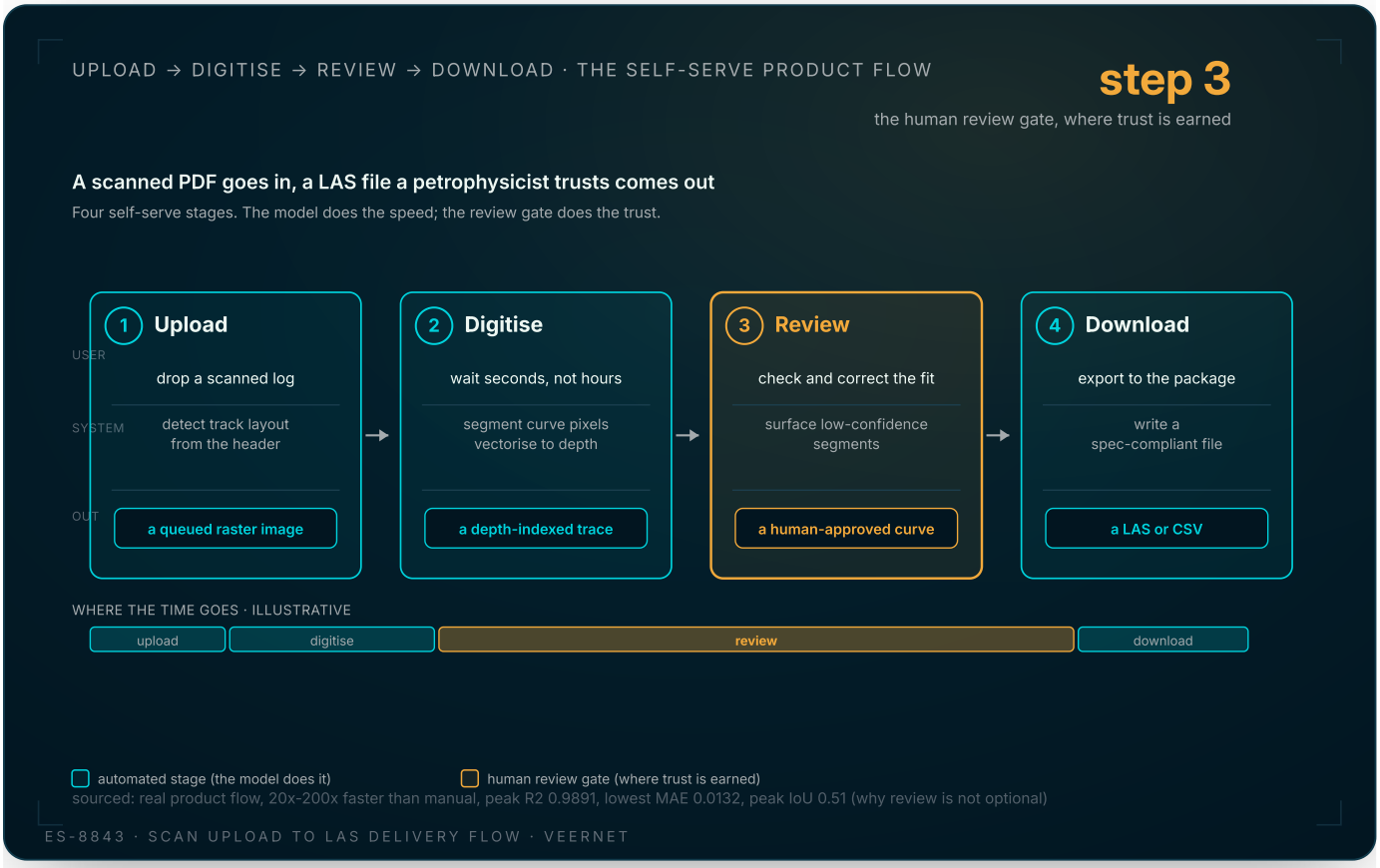
When we designed the service, we resisted the instinct to make it a thin wrapper over the model, because a thin wrapper would have shipped the model's imperfections straight to the user. Instead we designed the product as four explicit stages, and the model occupies only one of them. A user uploads a scanned raster log. The hosted model digitises it, segmenting the curve pixels and vectorising the mask into a depth-indexed trace. A human reviews the result and corrects what the model got wrong. The user downloads a spec-compliant LAS or CSV. Upload, digitise, review, download.

The reason this matters, and the reason the review stage cannot be an afterthought, is in the model's own numbers. VeerNet reaches a peak coefficient of determination of **0.9891** and a lowest mean absolute error of **0.0132**, which are excellent, but its peak curve-class intersection-over-union is only **0.51**. Those two facts are not in tension; they are the defining shape of a thin-structure digitiser. The model is right about most of the trace and wrong about a minority of it, and on a one-pixel-wide curve a small overlap error is geometrically large even when the recovered curve is substantially correct. A digitiser that is 99 percent right on the easy stretches and visibly wrong on a handful of crossings, faint segments, or label collisions is not a digitiser an operator will trust blind. It is a digitiser an operator will trust if, and only if, the product lets them see and fix exactly the parts the model got wrong. That is what the review stage is for. It converts a fast-but-imperfect prediction into a curve a petrophysicist will sign off on.

This is also why a generic image-vectorisation tool is not a substitute. Classical gridline-elimination and morphological approaches can extract clean curves from clean scans [5], but real archive scans are not clean: they have skew, bleed-through, overlapping traces, and handwritten annotations, and the failure modes of a classical tool on those inputs are

silent and hard to spot. A learned model paired with a review stage that surfaces low-confidence segments turns the failures from silent into visible, which is the property that makes the output trustworthy.

The second instrument renders the flow. It is four cards in a row, each showing what the user does, what the system does, and the artefact that leaves the stage, with connectors carrying the artefact forward. Three of the cards are teal, the automated stages where the model and the pipeline do the work. One card is orange, the review stage, because it is the only one that argues: it is where trust is earned, and it is the stage that separates a research demo from a service.



The productised digitiser as a four-stage, self-serve product flow. A user uploads a scanned raster log, the hosted model digitises it into a depth-indexed curve, a human reviews and corrects the imperfect prediction, and a spec-compliant LAS or CSV is downloaded. Each card shows what the user does, what the system does, and the artefact that leaves the stage. The orange stage is the only one that argues: the human-in-the-loop review gate, which exists precisely because the model digitises 20x to 200x faster than manual tracing and reaches a peak R-squared of 0.9891, yet its peak curve-class IoU is only 0.51. The review stage is what turns a fast-but-imperfect prediction into a curve an operator will sign off on, and it is the difference between a research demo and a service. The stage labels and artefacts are the real product flow; the per-stage timing band beneath the cards is an illustrative schematic of where time goes, not a measured benchmark.

Designing the review stage around an imperfect model

The hardest product decision in the whole service was how much to ask of the reviewer. Ask too little and the imperfections ship; ask too much and the speed advantage that justified the model evaporates, because a reviewer re-tracing every curve by hand is just manual digitisation with extra steps. The resolution was to make the review proportional to the model's uncertainty rather than uniform. The system surfaces the low-confidence segments, the crossings and faint stretches and ambiguous label regions where the model's prediction is least certain, and directs the reviewer's attention there, leaving the high-confidence majority of the trace untouched. A reviewer then spends their time on the minority of the curve that needs a human and skips the majority that does not.

That is why, in the instrument's illustrative timing band beneath the cards, the review segment is the widest. Upload and download are quick, digitisation is seconds of GPU time, and the human review is where the wall-clock time of a careful, trustworthy result actually goes. We are explicit that the band is illustrative geometry rather than a measured benchmark, but the shape is real and it is the shape that matters for product design: the model bought us the speed, and we spent some of that speed back on review to buy the trust. A service that pretended the model was perfect would have a faster timing band and a worse product. The review stage is the deliberate decision to be a little slower and a lot more trusted.

There is a second-order benefit we did not anticipate when we built it. Every correction a reviewer makes is a labelled example of a case the model got wrong, captured at exactly the point of failure. The review stage is therefore not only a quality gate; it is a data-collection mechanism that produces, for free, the hard examples that would most improve the next model. We did not close that loop in the first version of the service, and we note it in the limitations as work we left undone, but the architecture makes it possible, and that possibility is a direct consequence of having designed the review as a first-class stage rather than a bolt-on.

> **“A model with a peak overlap of 0.51 is not a weak model. It is a model that is right about most of a one-pixel curve, and the product's job is to make the minority it is wrong about easy to find and fix.”**

— FROM OUR REVIEW-DESIGN NOTES

III

THE PAYOFF

One model, many scans

Why the service is worth the engineering

Everything so far has been cost: the serving infrastructure, the four-stage flow, the review tooling, the file generation, the isolation and idempotency that a self-serve product demands. The payoff that justifies all of it is throughput, and specifically the throughput of one trained model run across many scans. This is the line that distinguishes a service from a one-off in the one dimension that compounds. A bespoke engagement digitises the scans in its scope and stops. A service points the same model at the next scan, and the next, and the cost of building the model is divided across all of them.

The acceleration figures make the payoff tangible. Against conventional manual workflows, deep-learning methods in upstream oil and gas accelerate by **20x to 200x** in the general case, with reservoir-engineering surrogates reaching **200x to 2000x** and production forecasting at least **100x** [4]. Our digitiser sits in the lower, general band, which is the conservative place to sit. Even there, the arithmetic of one model serving many scans is decisive. A batch that would take a person hundreds of hours to trace by hand is returned by the service in a small fraction of that, and the hours not spent tracing are hours returned to the petrophysicist for interpretation, the work only a person can do.

The third instrument is a throughput meter for exactly this. It shows one VeerNet model fanning out to many scans on the left, and on the right it computes, for an illustrative batch, the manual hours the work would have cost and the hours the service returns. The speedup lever spans the sourced **20x to 200x** range, and dragging it recomputes the dividend. The orange element, the hours-reclaimed bar, is the only one that argues, because the reclaimed hours are the entire economic case for the service rendered as a single quantity that grows with both the speedup and the number of scans the one model is pointed at, up to the **136,771** scans in the reference archive.

The productisation payoff as a throughput meter. One trained model, served to many scans, turns a manual tracing workload into machine seconds. Drag the speedup lever across the published acceleration range (20x to 200x faster than manual digitising) or use the arrow keys, and the instrument recomputes, for an illustrative batch of 500 scans at an illustrative 45 manual minutes each, the manual hours the same work would have cost and the hours the hosted service returns. The orange bar is the only one that argues: the hours reclaimed, the compounding dividend that exists only because the model is built once and run many times rather than rebuilt per project. The sourced facts are the 20x to 200x DNN acceleration range against conventional workflows (Koroteev and Tekic, 2021), one VeerNet model trained on 15,000 synthetic curves to a peak R-squared of 0.9891, and a reference archive of 136,771 TIF scans, the volume ceiling the one model can be pointed at. The batch size and the manual-minutes-per-scan baseline are illustrative inputs.

Reading the throughput meter honestly

Two things about the instrument deserve emphasis, one optimistic and one cautionary. The optimistic one is the shape of the curve. The hours reclaimed do not grow linearly with the speedup; they saturate. At 20x the service already returns the large majority of the manual hours, and pushing from 60x to 200x adds a smaller increment than the move from 1x to 20x did. This is good news for productisation, because it means the service does not need a record-breaking model to capture most of the available dividend. A conservative, trustworthy model at the low end of the sourced range already delivers nearly all the throughput benefit, which means the engineering effort is better spent on the service, the review, and the reliability than on squeezing the model from 60x to 200x.

The cautionary one is the inputs. The batch size and the manual-minutes-per-scan baseline in the instrument are illustrative, flagged as such, and the reclaimed hours are only as real as those inputs. The figures that are sourced are the speedup range, the model's accuracy, and the archive size; the batch economics are a teaching device for the relationship between speedup and hours, not a quoted result for any particular operator. We are deliberate about that line because the entire reason this piece exists is anonymisation and honesty, and the throughput dividend is exactly the kind of number that is easy to inflate. The honest claim is the shape: one model, many scans, with hours reclaimed that saturate quickly and scale with the archive. The dishonest claim would be a specific dollar figure, and we do not make it.

IV

THE BUILD

Packaging a research model for serving

What had to change in the model's surroundings

The model itself barely changed between the engagement and the service. The weights are the same, the architecture is the same: a five-stage encoder of stride-two convolutions, a bottleneck refined by **2** transformer attention layers [7], and a five-stage decoder that upsamples back to a full-resolution mask in the encoder-decoder tradition [6]. What changed was everything the research notebook had been allowed to ignore.

In the notebook, an inference was a function call on a curated image with a person watching. In the service, an inference is a request from an unknown user with an arbitrary scan, possibly malformed, possibly enormous, possibly a duplicate of a request already in flight. The serving layer had to acquire the model once and hold it warm rather than loading weights per request, because cold loading on every upload would make the latency unacceptable and the cost absurd. It had to handle the variable dimensions of real scans, which range widely in width and height, without falling over on an image larger than anything in training. It had to be idempotent, so that a user who retries an upload does not pay twice or get two divergent results. And it had to fail gracefully, returning a clear signal on an input the model cannot handle rather than a confident wrong answer, because a confident wrong answer on a self-serve product is worse than an honest refusal.

None of this is exotic. It is the standard distance between a model that works in a notebook and a service that works in production, and the software-engineering-for-machine-learning literature documents it as a recognised discipline with its own practices, distinct from both conventional software engineering and from model research [2]. The reason we dwell on it is that it is the part teams consistently underestimate. The model took a known, bounded effort; the engineering around it to make a self-serve service trustworthy was the larger and less predictable half, and pretending otherwise is how productisation projects overrun.

Generating a file someone else's software has to read

A detail that turned out to carry more weight than expected was the output format. In the engagement, we delivered LAS and CSV files that we generated and checked ourselves, so small deviations from the standard were caught by eye before handover. In the service, the file is generated automatically and downloaded by a user whose petrophysics package will reject anything that is not exactly to spec. A spec-compliant LAS has a precise header structure, a correctly declared depth index, a curve-information section that matches the data, and consistent null handling, and a package that reads it is unforgiving of any departure.

So the file generation had to move from something a person validated to something the system guaranteed. We built the export so that the depth-indexed trace coming out of the review stage is serialised into a LAS that conforms to the standard by construction, with the header, the curve declarations, and the data section assembled from the trace rather than hand-edited. The same trace can be emitted as a CSV for users who want the raw numbers without the well-log structure. This is unglamorous work, and it is exactly the kind of work that does not exist in a one-off because a person does it by hand, and that becomes load-bearing in a service because no person is in the loop. The deployment literature is full of services that produced technically correct model outputs that downstream systems could not consume [3]; the file generation is where we made sure we were not one of them.

Cost and the posture it forces

The shift from per-project to per-scan cost changes the posture toward compute in a way worth stating. In the engagement, GPU cost was a line item in a fixed budget, and at monthly rates of roughly **EUR 750** for a high-end card and **EUR 1,800** for an advanced one, it was a small and predictable fraction of a project priced in the hundreds of thousands. In the service, GPU cost is a variable that scales with usage, and it sits directly against the value of each scan. That does not make the service expensive; the per-scan cost in GPU seconds is tiny against the manual alternative. But it does mean the service has to be conscious of compute in a way the engagement never was: holding the model warm without burning idle capacity, batching where possible, and sizing the serving tier to demand rather than to a project budget. The amortisation that makes the service attractive only works if the serving cost is kept proportional to actual use, which is its own small discipline.

V

THE RESULTS

What the productised service is, in numbers

The model carried over intact

The first result is that nothing about the model regressed in the move to serving. The accuracy that was achieved in the engagement is the accuracy the service delivers, because the weights are unchanged: a peak coefficient of determination of **0.9891**, a lowest mean absolute error of **0.0132**, and the **20x to 200x** speedup over manual tracing [4]. That continuity is itself a result worth stating, because a common failure of productisation is silent degradation, where the served model behaves differently from the validated one due to preprocessing drift or input handling. We held the preprocessing identical between training and serving precisely to avoid that, and the served model's outputs match the validated model's on the same inputs.

The second result is the shape of the trade the service makes between speed and trust. The model gives back the speed; the review stage spends a measured amount of it to buy a result an operator will sign off on. The peak curve-class intersection-over-union of **0.51** is not a number we hide; it is the number that justifies the review stage, and the service is honest that the model finds most of the trace and the human finishes the minority. A service that quoted only the 0.9891 and omitted the 0.51 would be overselling; a service that quoted only the 0.51 would be underselling. Both are true, and the product is designed around their coexistence.

The economics carried over inverted

The third result is the one the whole effort was for: the economics inverted. The bespoke engagement amortised the model build across one project; the service amortises it across every scan every operator uploads, against a reference archive of **136,771** scanned images. The marginal cost of the next scan fell from a new project to GPU seconds. The throughput

dividend, illustrated in the third instrument, is real in shape even where the specific batch figures are illustrative: hours reclaimed that saturate quickly with the speedup and scale with the number of scans. This is what a service buys that a one-off cannot. The model was the same in both worlds; the value the model created was multiplied by the number of times it could be run, and productisation is precisely the work of making that number large.

THE SAME MODEL, PRODUCTISED

0.9891

Peak R-squared, carried over intact

0.51

why review exists

Peak curve-class IoU

GPU seconds

inverted

Marginal cost per scan, down from a project

1 build

136,771 scans

Amortised across the archive

VI

THE ROADMAP

What productisation opens up next

Reading the three instruments as one argument

The three instruments are meant to be read in sequence, because together they are the whole case. The ledger is the why: it shows that the bespoke and hosted shapes run the same model and differ only in the dimensions that compound, and it names amortisation as the property that makes a service worth building. The flow is the how: it shows the four self-serve stages and isolates the review gate as the load-bearing element that turns an imperfect prediction into a trusted file. The throughput meter is the payoff: it shows one model serving many scans and the hours that returns, the dividend that justifies the engineering. Read in that order, they trace the productisation decision from the shape change, through the product design, to the economic result.

The work we would do next

Productising the digitiser opened a set of next steps that the one-off never reached, and naming them is part of being honest about where the service stands. The most valuable is closing the review loop: the corrections reviewers make are labelled hard examples captured at the point of failure, and feeding them back into training would steadily reduce the share of the trace the model gets wrong, which would shrink the review burden, which would raise the throughput, a virtuous loop the architecture supports but the first version does not yet run. Beyond that lies multi-tenant isolation hardened for operators who are competitors and must never see each other's scans, batch upload for operators who want to digitise an archive rather than a scan at a time, and confidence reporting on the downloaded file so a petrophysicist knows which segments the model was sure about and which the reviewer touched.

None of these require a better model, and that is the recurring lesson of the whole exercise. The model was the part we could write a paper about, and it was the easy half. The service, the flow, the review, the file generation, the isolation, the cost discipline, and the loop back into training are the half that decides whether a capability built for one operator ever reaches a second. We built a model that digitised one archive; we are building a service so that the next archive does not need us, and the difference between those two sentences is the entire content of this whitepaper.

WHAT CARRIES FROM THIS PRODUCTISATION

1. A research model is the easy half. The same VeerNet weights run in both the bespoke engagement and the hosted service; everything that changed was the serving, flow, review, and file generation around the model, which is where the cost and the risk live.
2. The shape change inverts the economics. A one-off amortises the build across one project; the service amortises it across up to **136,771** scans, dropping the marginal cost of the next scan from a new project to GPU seconds.
3. The review stage is load-bearing, not optional. A peak curve-class IoU of **0.51** against a peak R-squared of **0.9891** means the model is right about most of a one-pixel curve and wrong about a minority, and the product must make that minority easy to find and fix.
4. The throughput dividend saturates early. Most of the reclaimed hours arrive by the low end of the sourced **20x to 200x** range, so effort is better spent on the service and the review than on pushing the model speedup higher.
5. Self-serve generates its own training data. Every reviewer correction is a labelled hard example at the point of failure, a loop the service architecture supports and the next version should close.

Limitations

This account carries real boundaries, and stating them is part of making the productisation story trustworthy rather than promotional.

The throughput figures in the third instrument rest on illustrative inputs. The speedup range of 20x to 200x, the model accuracy, and the archive size are sourced; the batch of 500 scans and the 45-minute manual baseline per scan are flagged illustrative parameters chosen to make the relationship between speedup and reclaimed hours legible. The reclaimed-hours quantity is therefore a shape, not a quoted result for any operator, and it should be read as the structure of the dividend rather than a number to put in a budget. We made this choice deliberately, because an inflated specific figure is exactly the kind of claim this anonymised piece exists to avoid.

The review loop is described as an opportunity, not a shipped feature. The architecture captures reviewer corrections as labelled examples, but the first version of the service does not feed them back into training, so the virtuous loop between review effort and model improvement is a roadmap item rather than a measured result. Claiming the loop as operational would overstate where the service stands.

The accuracy numbers are the engagement's validation results, carried into the service by holding the weights and preprocessing fixed. The peak R-squared of 0.9891 was reached on the hardest curve in the validation set, and the peak curve-class IoU of 0.51 is a validation figure as well; both describe model behaviour measured during the engagement, not a continuously monitored production metric. A mature service would track served-input accuracy over time and watch for drift, and the first version reports the validated numbers rather than a live production distribution.

Finally, the service is specific to the deliverable and the input. It digitises scanned raster well logs into LAS and CSV, and the productisation decisions, the review design, the file generation, and the throughput case are shaped by that specific task. The general lesson, that the model is the easy half and the service is the deciding half, is portable; the particular flow and the particular numbers are not a template for every model-to-service transition, and we claim the method, not a universal blueprint.

GLOSSARY

Amortisation

Spreading a fixed cost across many uses. A bespoke model is built once and used once, so its build cost falls on one project; a hosted model is built once and used across many scans, so its build cost is divided by the number of scans served.

Bespoke deliverable

A fixed-scope project that produces a defined set of artefacts for one client and is billed once. The original VeerNet engagement was bespoke: we digitised a named set of scans and handed the LAS and CSV files over at the end.

Hosted service

Software run on infrastructure we operate, reached over the network, where the user brings the input and the system returns the output. The productised digitiser is a hosted service: the user uploads a scan and downloads a file, and the model lives on our side.

Human-in-the-loop review

A stage where a person inspects and corrects the model's output before it is accepted. It exists because the model is accurate but imperfect, and it is the difference between shipping a fast prediction and shipping a curve an operator will sign off on.

Productisation

Turning a one-off capability into a repeatable product. For the digitiser it meant taking a model that ran inside an engagement and wrapping it in a hosted service that any operator can use themselves, with the upload, serving, review, and download all handled without the original team present.

Self-serve

A product a user can complete without a human operator on the vendor side. Self-serve digitisation means the

	operator drives the upload to download flow themselves, which is the property that makes one model able to serve many users at once.
Serving	Running a trained model to answer requests in production, as opposed to training it. Serving introduces concerns the research notebook never had: latency, concurrency, idempotency, cost per request, and graceful failure on inputs outside the training distribution.
Spec-compliant LAS	A Log ASCII Standard file that conforms to the format a petrophysics package expects, with a correct header, depth index, and curve sections. The deliverable of the service is a spec-compliant LAS or CSV, not a raw model output.
Throughput dividend	The cumulative tracing hours returned when one trained model is run across many scans instead of a person tracing each one. It scales with the model speedup and the number of scans, and it is the payoff that justifies the productisation effort.

References

1 Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., Dennison, D. (2015). *Hidden Technical Debt in Machine Learning Systems*. NeurIPS. https://papers.nips.cc/paper/2015/hash/86df7dcfd896fcaf2674f757a2463eba-Abstract.html

2 Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., Zimmermann, T. (2019). *Software Engineering for Machine Learning: A Case Study*. ICSE-SEIP. https://www.microsoft.com/en-us/research/publication/software-engineering-for-machine-learning-a-case-study/

3 Paleyes, A., Urma, R.-G., Lawrence, N. D. (2022). *Challenges in Deploying Machine Learning: A Survey of Case Studies*. ACM Computing Surveys. https://dl.acm.org/doi/10.1145/3533378

4 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

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/s13202-019-0670-5

6 Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H. (2018). *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation*. ECCV. https://arxiv.org/abs/1802.02611

7 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

Get the full whitepaper

This page is the long-form summary. The complete whitepaper adds the serving architecture in full, the idempotency and isolation design for a multi-tenant self-serve flow, the LAS export conformance checklist, the review-stage uncertainty surfacing in detail, and the worked throughput model behind the third instrument.

EARTHSCAN

From One-Off to On-Demand: Productising a Raster-Log Digitiser into a Hosted Scan-to-LAS Service

Published February 2025. Generated 2026-07-22.
https://www.earthscan.io/whitepapers/productising-raster-log-digitisation-bespoke-to-hosted