DOCUMENTO DE DEFINICIÓN

Subnet de Especialización y Destilación Competitiva de Modelos de Vídeo Generativo

Motivo, actividad, utilidad, diseño de validación y evolución del protocolo

TESIS CENTRAL

La subnet convierte distribuciones repetitivas de trabajo de vídeo en modelos especialistas más baratos, rápidos y fiables. Los miners compiten por entrenar y comprimir esos modelos; los validators comprueban su calidad, generalización, coste y legitimidad sobre datos ocultos.

Versión 1.0
26 de agosto de 2026
Estado: definición propuesta para MVP y evolución posterior

Nota de alcance. Este documento define una dirección de producto y protocolo. Las referencias legales son criterios de diseño y no sustituyen una revisión jurídica específica de cada modelo, dataset, teacher, territorio o contrato de cliente.

Contenido

Ficha de definición

Elemento	Definición
Propósito	Transformar workloads repetitivos de vídeo en modelos especialistas con menor coste por resultado aceptado.
Unidad de trabajo	Un challenge vertical y medible, definido mediante un TaskSpec.
Actividad del miner	Entrenar, especializar, destilar, comprimir y empaquetar un modelo reproducible.
Actividad del validator	Ejecutar prechecks y evaluar en inputs ocultos la calidad, generalización, eficiencia, integridad y procedencia.
Output de la subnet	Modelo o adapter, configuración de inferencia, container, lineage, licencia, benchmark y perfil de coste.
Métrica económica	Coste total por vídeo aceptado, no precio por segundo ni coste bruto por intento.
Fase inicial	Competición por modelos; no inference marketplace, router ni producto creativo completo.
Evolución	Registry de especialistas, competición por pipelines, routing y serving cuando exista demanda real.

1. Resumen ejecutivo

Los modelos frontier de vídeo se diseñan para cubrir una enorme variedad de escenas, estilos, movimientos, personajes y condiciones. Esa generalidad es valiosa para el long tail creativo, pero es económicamente ineficiente cuando una empresa repite una transformación estrecha cientos de miles de veces: animar un producto, ejecutar un movimiento de cámara, añadir un efecto, crear un loop o mantener una identidad concreta.

La subnet propuesta crea un mercado competitivo de investigación y entrenamiento. Cada challenge describe una distribución de trabajo concreta. Los miners compiten por producir el modelo especialista que la resuelve con la mejor relación entre calidad, fiabilidad y coste. Los validators no premian el vídeo más llamativo de una demo, sino el modelo que generaliza a inputs no vistos, cumple restricciones duras y minimiza el coste por asset aceptado.

La subnet no empieza como una red de inferencia. En su primera versión, la actividad útil es fabricar y seleccionar modelos. El serving puede mantenerse fuera de la subnet hasta que descentralizarlo aporte una ventaja demostrable. De este modo se evita confundir tres negocios distintos: entrenamiento competitivo, serving de GPU y experiencia de producto.

DEFINICIÓN OPERATIVA

La subnet no genera vídeos como actividad principal. Genera modelos que permiten producir determinados vídeos de forma más económica y controlada.

El protocolo es horizontal, porque puede aceptar diferentes familias de workloads. Cada challenge es vertical, porque solo una tarea estrecha permite entrenar, medir y comparar de forma objetiva. La visión de “fábrica de modelos” solo queda validada cuando el protocolo demuestra que puede repetir este proceso en varios challenges y, posteriormente, evaluar pipelines capaces de crear especialistas ante tareas nuevas.

2. Definición y propósito de la subnet

2.1. Definición

La subnet es un protocolo de competición descentralizada para la especialización, destilación y compresión de modelos generativos de vídeo. Recibe una especificación de workload y convierte la capacidad de investigación de los miners en artefactos evaluables y reutilizables.

El objetivo formal es seleccionar un modelo M que minimice el coste total por resultado aceptado, sujeto a un umbral de calidad, cumplimiento y reproducibilidad:

OBJETIVO DE OPTIMIZACIÓN

Champion = arg min C_total_por_aceptado(M), sujeto a Q(M) ≥ Q_mínima, generalización válida, ejecución reproducible y procedencia legal aprobada.

La palabra “destilación” se utiliza en un sentido amplio de producto. El resultado puede venir de step distillation, consistency o distribution matching, pero también de LoRA, fine-tuning, cuantización, sparse attention, caching, cambios de sampler o students arquitectónicamente distintos. La subnet debe optimizar el resultado económico, no imponer una única receta académica.

ACLARACIÓN TÉCNICA

Un modelo económicamente “más pequeño” no tiene por qué tener menos parámetros. También puede mantener el tamaño y reducir de forma radical el número de pasos, las evaluaciones de classifier-free guidance, la precisión numérica, la memoria o la latencia.

2.2. Propósito económico

La utilidad aparece cuando un workload tiene volumen suficiente para amortizar el coste de crear un especialista. La subnet debe convertir una repetición de demanda en una ventaja de coste o fiabilidad que pueda medirse fuera del protocolo.

Una aproximación útil al coste es:

COSTE POR RESULTADO ACEPTADO

C_aceptado ≈ (coste de inferencia por intento + QA y operación por intento) / probabilidad de aceptación.
C_total ≈ C_aceptado + coste de especialización / número esperado de assets durante la vida útil del modelo.

Esta métrica evita una conclusión engañosa: un endpoint de cinco céntimos no es barato si obliga a generar seis variantes. Un especialista de doce céntimos puede ser superior si acierta al primer intento y preserva identidad, composición y movimiento.

2.3. Propósito de protocolo

La subnet organiza la búsqueda sobre un espacio de soluciones demasiado amplio para una única receta central. Diferentes miners pueden aportar datos, teachers abiertos, students, arquitecturas, schedules y optimizaciones distintas. El validator transforma esa diversidad en una competición verificable y orientada a resultados.

3. Problema que resuelve

3.1. Generalidad innecesaria

Un modelo generalista debe mantener capacidades para una distribución abierta de prompts: composición, estilos, cámara, física, anatomía, narración, audio, múltiples sujetos y cambios de escena. Un workload empresarial suele utilizar una fracción pequeña y repetitiva de esa capacidad. La tesis consiste en dejar de pagar por la cobertura completa cuando el trabajo real es estrecho.

3.2. Repetición de workflows, no de frases

La repetición relevante no es necesariamente textual. Dos prompts diferentes pueden representar la misma transformación operativa: “orbita este perfume”, “mueve la cámara alrededor de esta botella” y “crea un product shot con arco lateral” pertenecen al mismo cluster. La capa de demanda debe agrupar intenciones, assets y restricciones, no limitarse a buscar strings idénticos.

3.3. El precio por segundo oculta los fallos

Los proveedores suelen expresar precios por segundo o por generación. Para una operación real importan también los reintentos, la revisión, la edición, la latencia, la preservación del sujeto y la tasa de rechazo. La subnet debe medir el coste del output utilizable, no el coste del primer MP4 producido.

3.4. La calidad visual genérica no basta

Benchmarks como VBench separan la calidad en dimensiones como consistencia de identidad, suavidad de movimiento, flicker y relaciones espaciales. VBench-2.0 amplía la evaluación hacia controlabilidad, física y commonsense. Esto refuerza una conclusión de diseño: “vídeo bonito” es una métrica demasiado agregada para seleccionar un modelo comercialmente fiable. [Ref. 7]

Cada challenge debe definir qué significa éxito en su propio dominio. Para un packshot pueden dominar OCR, geometría y trayectoria de cámara. Para interiores importan paredes, puertas y perspectiva. Para un loop de personaje importan pose, periodicidad y drift. El benchmark del nicho es parte del producto de la subnet. [Ref. 8]

3.5. El gap abierto no invalida el enfoque

Los modelos abiertos no necesitan superar al frontier en todo. Solo necesitan poseer capacidad base suficiente dentro del workload elegido. Wan 2.2 TI2V-5B, por ejemplo, soporta text-to-video e image-to-video a 720p y 24 fps y puede ejecutarse en una RTX 4090; su licencia Apache 2.0 facilita la creación y distribución de derivados. Ese perfil lo convierte en un baseline razonable para un track accesible, no en una garantía de calidad generalista. [Ref. 3]

La viabilidad técnica de la compresión ya está demostrada en el ecosistema: HunyuanVideo 1.5 publica un checkpoint step-distilled de 8/12 pasos con una reducción relevante de tiempo en una RTX 4090, mientras Cosmos 3 publica una variante I2V de cuatro pasos que sustituye un schedule mucho más costoso, aunque sobre hardware de gran escala. La oportunidad de la subnet no es inventar la destilación, sino orientarla a workloads y economía reales. [Ref. 4] [Ref. 5]

4. Tesis de funcionamiento y condiciones de utilidad

4.1. Hipótesis central

Una colección de modelos estrechos puede absorber una parte creciente de los workloads repetitivos que hoy se resuelven con modelos frontier innecesariamente generales. El long tail continúa usando un fallback generalista; los clusters frecuentes migran a especialistas.

4.2. Condiciones necesarias

Condición	Por qué es necesaria	Señal mínima
Capability overlap	El modelo base ya debe saber aproximar la tarea; la destilación no crea capacidades inexistentes.	Aceptación no trivial del baseline en el nicho.
Repetición y volumen	El ahorro debe amortizar entrenamiento, evaluación e integración.	Frecuencia, coste actual y vida útil medibles.
Evaluabilidad	Sin métricas y held-out fiables, los miners optimizan una apariencia fácil de manipular.	Gates automáticos y evaluación humana alineada.
Derechos claros	Un champion no tiene utilidad si no puede entrenarse, distribuirse o servirse legalmente.	Allowlist de modelos y ProvenanceManifest.
Espacio de búsqueda	La subnet aporta valor si hay múltiples estrategias plausibles y no una receta obvia.	Diversidad real de submissions.
Demanda externa	Las emisiones no deben ser el único comprador del resultado.	Design partner, API, licencia o challenge privado.

4.3. Cuándo no sería útil

- Si el modelo abierto falla casi siempre incluso antes de especializarlo.
- Si una plantilla, un pipeline 2.5D o una solución 3D es mucho más fiable y barata.
- Si el cluster tiene poco volumen o una vida útil demasiado corta para amortizar el entrenamiento.
- Si la evaluación es esencialmente subjetiva y fácil de Goodhart.
- Si los miners solo ejecutan el mismo notebook y no aportan investigación diferenciada.
- Si no existe un uso externo de los modelos producidos.

PRINCIPIO DE HONESTIDAD

La subnet es una hipótesis validable. Su razón de existir debe demostrarse con ahorro, aceptación y generalización, no únicamente con una landing o una demo seleccionada manualmente.

5. Alcance inicial y límites

5.1. Qué hará en V0

- Publicar challenges estrechos con un contrato de entrada, salida, calidad, hardware y licencia.
- Aceptar commitments de modelos, adapters y configuraciones reproducibles.
- Ejecutar los candidatos en un entorno controlado y sin acceso de red.
- Medir calidad, generalización, coste, latencia, VRAM, diversidad e integridad.
- Seleccionar un champion y mantener un registry versionado de modelos especialistas.
- Convertir las mejoras en evidencia pública y, cuando corresponda, en activos comercializables.

5.2. Qué no hará al inicio

Fuera de alcance inicial	Motivo
Serving descentralizado tipo marketplace de GPU	Añade disponibilidad, batching, colas y verificación online antes de validar el modelo.
Router generalista o suite creativa	No tiene sentido enrutar entre especialistas que todavía no existen.
Foundation model propio de frontera	Requiere datasets y compute fuera del wedge inicial.
Text-to-video completamente abierto	Demasiada entropía para un primer challenge objetivo y barato.
Distillation desde APIs cerradas por defecto	Puede contravenir términos contractuales y limita la reproducibilidad.
Promesa de fábrica automática universal	V0 demuestra una arena; la fábrica requiere varios challenges y evaluación de pipelines.

5.3. Modalidad recomendada

Image-to-video es la modalidad preferente para el MVP. La imagen ancla sujeto, color y composición, reduce la entropía y permite evaluar preservación con más objetividad. Text-to-video puede incorporarse en challenges posteriores cuando el protocolo y el evaluator estén maduros.

6. Actores y actividad de la subnet

Actor	Actividad principal	Valor aportado
Sponsor u operador del challenge	Aporta o define el workload, sus restricciones, baseline, datos y presupuesto de evaluación.	Conecta el protocolo con demanda verificable.
Miner	Entrena, especializa, destila, comprime y empaqueta un candidato.	Explora el espacio de soluciones y produce propiedad intelectual útil.
Validator	Prevalida, ejecuta, mide, audita y publica resultados.	Convierte outputs estocásticos en una competición confiable.
Registry	Mantiene lineage, hashes, licencias, scores, hardware y versiones.	Hace reutilizables y auditables los activos producidos.
Capa comercial futura	Sirve champions, enruta tráfico y recoge feedback.	Transforma el ahorro técnico en uso e ingresos externos.

6.1. Actividad del miner

El miner recibe un TaskSpec y decide cómo resolverlo dentro de una allowlist técnica y legal. Puede generar datos sintéticos permitidos, seleccionar ejemplos, entrenar LoRAs o adapters, realizar fine-tuning completo, aplicar distillation, cuantizar, cambiar la arquitectura del student o modificar el schedule de inferencia. El protocolo define la interfaz y el objetivo; no prescribe una receta única.

La submission mínima debe contener:

- Weights, adapter o checkpoint inmutable y versionado.
- Container y script de inferencia reproducibles.
- Configuración de sampler, pasos, precisión, seed policy y requisitos de hardware.
- Lineage del modelo base y de los teachers utilizados.
- ProvenanceManifest de datos y derechos.
- Licencia de la submission y obligaciones downstream.
- Model card con limitaciones y casos no soportados.

6.2. Actividad del validator

El validator no entrena el modelo y no acepta métricas autodeclaradas. Descarga o recibe la revisión comprometida, verifica hashes y licencias, ejecuta el container sobre hardware estándar y genera outputs con inputs ocultos y múltiples seeds. Publica una matriz de resultados y un score reproducible.

6.3. Actividad temporal

A diferencia de una subnet de inferencia, la actividad no se mide en requests por minuto. Se organiza en ciclos de challenge: apertura, ventana de entrenamiento, commit, evaluación, challenge al incumbent, coronación y mantenimiento del champion. Los ciclos pueden durar días o semanas según el coste del entrenamiento y la evaluación.

7. Ciclo de vida de un challenge

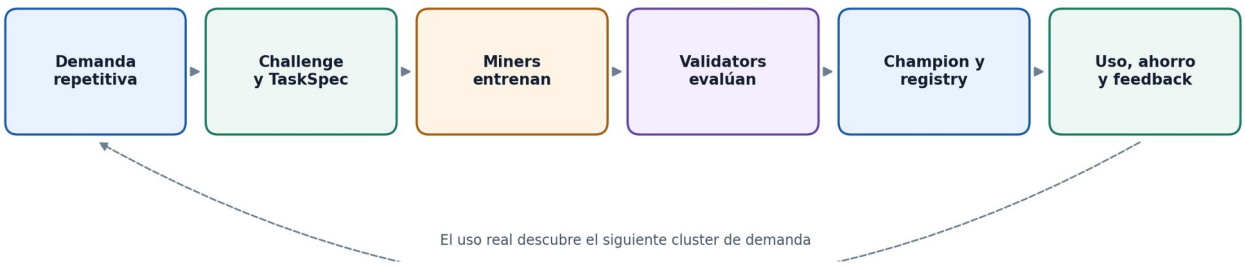

Figura 1. El protocolo transforma demanda repetitiva en un modelo especialista y utiliza el feedback para definir el siguiente challenge.

Descubrimiento de demanda. Se agrupan prompts, assets, resultados aceptados y rechazados, frecuencia, coste y SLA. El cluster debe tener volumen y coherencia operacional.

TaskSpec. Se fija modalidad, contrato de inputs, output, restricciones, calidad mínima, hardware, budget, licencias y política de datos.

Dataset y hidden set. Se prepara training data permitido y un test oculto con identidades, materiales, textos, composiciones y seeds no vistos.

Baseline. Se publica un modelo base reproducible y, cuando sea útil, una alternativa no generativa para medir el valor real del enfoque.

Ventana de minería. Los miners entrenan y suben artefactos inmutables.

Commit y prechecks. Se registra revisión, hash, licencia, tamaño, arquitectura y container; los inválidos se filtran sin GPU.

Evaluación oculta. Se ejecutan múltiples seeds, métricas específicas, pairwise tests y revisión humana.

Selección. Solo los candidatos que superan los gates de calidad compiten en coste por aceptado y eficiencia.

Registry y explotación. El champion se registra, se prueba en tráfico real y genera feedback para el siguiente ciclo.

SEPARACIÓN CRÍTICA

El training set puede ser parcialmente público. El hidden set no debe serlo. El challenge evalúa generalización; no premia la memorización de una lista fija de productos o prompts.

8. Sistema de validación

8.1. Principio de gates

La validación debe separar calidad mínima y eficiencia. Si ambas se mezclan desde el primer punto, un modelo muy barato pero inútil puede compensar sus fallos con coste. El sistema recomendado funciona por etapas eliminatorias.

Gate	Qué verifica	Resultado
0. Procedencia e integridad	Licencia, lineage, derechos, hash, revisión, duplicados, container y ausencia de llamadas externas.	Descalificación si falla.
1. Ejecución	Formato, duración, resolución, determinismo operativo, VRAM y timeout.	Descalificación o penalización dura.
2. Calidad mínima	Éxito de la transformación, preservación, temporalidad, prompt adherence y seguridad.	Elegible o no elegible.
3. Generalización	Rendimiento en identidades, materiales, textos y composiciones no vistas.	Evita overfitting al challenge público.
4. Eficiencia	Coste por intento, latencia, VRAM, pasos y coste por aceptado.	Ordena a los candidatos válidos.
5. Robustez estadística	Variación entre seeds, failures, paired re-evaluation y margen de dethronement.	Evita cambios por ruido.

8.2. Métricas específicas del workload

Cada challenge utiliza una combinación distinta. Un VLM general puede participar, pero nunca debe ser el único juez. Los evaluadores generales son vulnerables a reward hacking y pueden preferir estética sobre fidelidad.

Dimensión	Ejemplos de métricas
Éxito de tarea	Clasificador de transformación, trayectoria estimada, pose, depth o acción objetivo.
Preservación	DINO/embedding, segmentación, bordes, color delta, OCR y edit distance de texto.
Temporalidad	Optical flow, flicker, consistencia de fondo, drift y periodicidad.
Adherencia	VLM task-specific, comprobaciones de objetos añadidos o eliminados y constraints.
Diversidad	Distancia entre seeds y ausencia de mode collapse.
Aceptación humana	¿Se utilizaría sin regenerar? Pairwise preference y defectos descalificantes.
Eficiencia	Tiempo end-to-end, peak VRAM, energía o GPU-seconds y tasa de fallo.

8.3. Coste por aceptado

La tasa de aceptación debe calcularse sobre una política de sampling fija. No es válido permitir que un modelo entregue su mejor resultado de cincuenta intentos mientras otro se mide con uno. Cada candidato recibe el mismo número de seeds y el mismo budget.

Una forma simple de ranking es: primero superar Q_mínima; después minimizar C_total_por_aceptado. Los tie-breakers pueden ser calidad residual, latencia, VRAM y diversidad. La frontera de Pareto debe conservarse como telemetry aunque exista un único champion.

8.4. Evaluación humana

El panel humano se utiliza para calibrar y auditar las métricas automáticas. No debería revisar cada output para siempre, pero sí una muestra estadísticamente suficiente, todos los potenciales dethronements y los casos donde el evaluator automático presenta baja confianza.

- ¿Sigue siendo exactamente el mismo sujeto o producto?
- ¿Cumple la transformación solicitada?
- ¿Existe un fallo que impida usar el asset?
- ¿Se utilizaría el resultado sin regenerarlo?
- ¿Cuál de dos outputs equivalentes es preferible?

8.5. Anti-gaming

- Prompts, assets y variaciones procedimentales generados a partir de seeds impredecibles antes del commit.
- Revision pinning y hashing de weights, adapters y containers.
- Detección de duplicados y near-copies mediante fingerprints funcionales.
- Entorno no-egress para impedir proxying de APIs externas.
- Canary inputs y tests de contaminación para detectar memorización.
- Reevaluación emparejada del incumbent y el challenger sobre el mismo batch.
- Margen mínimo de dethronement y confidence intervals para controlar el ruido estocástico.

La experiencia de SN97 es relevante: sus evaluaciones evolucionaron desde métricas más simples hacia múltiples ejes, prompts procedimentales, re-evaluación del King, fingerprints y mecanismos contra overfitting. En vídeo, donde cada sample es más caro y variable, estas defensas deben diseñarse desde el inicio. [Ref. 1]

9. Selección del champion e incentivos

9.1. Champion

El champion es el modelo válido que ocupa la mejor posición económica para el hardware y SLA definidos. La coronación no significa “mejor modelo de vídeo”; significa “mejor solución verificada para este TaskSpec y esta versión del benchmark”.

9.2. Pareto frontier

La subnet debería conservar más de un perfil útil: un champion ultraeficiente en 24 GB, un champion premium para hardware de datacenter o un modelo con mayor calidad y menor velocidad. El registry puede reflejar esos perfiles aunque los pesos on-chain requieran una regla más simple.

9.3. Principios de incentivos

- No recompensar GPU-hours ni volumen de training por sí mismos.
- No pagar submissions que no superen calidad, integridad y procedencia.
- Asignar la mayor recompensa al champion, pero reservar una fracción a challengers válidos o a la frontera de Pareto durante la fase inicial.
- Exigir disponibilidad del artefacto durante un periodo mínimo y revalidar su integridad.
- Separar eventualmente recompensas por producción de modelos y por serving; son actividades y costes distintos.

NOTA DE DISEÑO

Un winner-take-all puro es fácil de comunicar, pero puede reducir la participación cuando entrenar vídeo es caro y la evaluación es ruidosa. Para el MVP conviene un esquema híbrido: champion dominante, rewards menores para challengers que superen el baseline y fondos explícitos para exploración.

10. Utilidad económica y activos producidos

10.1. Para clientes y plataformas

La propuesta comercial asociada es: "entrega tu distribución repetitiva de vídeo y recibe un modelo optimizado para ella". El cliente aporta volumen, inputs, outputs aceptados y rechazados, coste actual, SLA y restricciones. La subnet devuelve un specialist artifact, benchmark, perfil de coste y configuración de despliegue.

10.2. Para miners

Los miners monetizan investigación aplicada: generación y selección de datos, entrenamiento, destilación, optimización de inferencia y diseño de students. La ventaja competitiva no es mantener una GPU ociosa, sino producir un artefacto que supera a los demás bajo el mismo contrato.

10.3. Para el ecosistema

Cada challenge genera más que weights. Produce un benchmark privado, una curva de coste-calidad, conocimiento sobre técnicas efectivas, un dataset con derechos trazables, un historial de modelos y una reputación por miner. El evaluator y el registry pueden convertirse en activos tan importantes como el checkpoint.

10.4. Vías de monetización futuras

Vía	Qué se vende	Relación con la subnet
Challenge privado	Creación competitiva de un especialista para una empresa.	La demanda financia una tarea y sus rewards.
Licencia	Derecho de uso del champion o de un perfil de la Pareto frontier.	El registry gestiona lineage y condiciones.
API de inferencia	Uso por generación o por asset aceptado.	Puede servirse centralmente al inicio.
Router de especialistas	Selección automática de specialist o fallback generalista.	Aparece cuando existe catálogo y tráfico.
Optimización continua	Reentrenamiento con feedback y nuevos inputs.	Genera nuevos challenge rounds.

10.5. Activo defensivo

El moat no es un único checkpoint. Es el bucle completo de demanda, clustering, datos, training, benchmark, serving y feedback. Los weights pueden copiarse; es más difícil replicar el historial de aceptación, los hidden sets, la comunidad de miners y el acceso a workloads reales.

11. Justificación de la descentralización

Para el primer modelo, un equipo centralizado será probablemente más rápido. La subnet se justifica cuando el problema se convierte en una secuencia de búsquedas abiertas donde múltiples equipos pueden explorar soluciones distintas y el resultado puede evaluarse objetivamente.

La subnet aporta valor cuando…	No lo aporta cuando…
Hay una cola de challenges y workloads externos.	Solo se entrenará un modelo una vez.
Existen muchas recetas plausibles y no se conoce la mejor.	La solución es obvia y todos ejecutan el mismo script.
Los miners tienen datos, técnicas o arquitecturas diferenciadas.	La única diferencia es quién alquila más GPU.
El validator puede medir outputs sin confiar en el miner.	La calidad es totalmente subjetiva o no auditable.
Los champions se utilizan y generan ahorro fuera de la red.	Las emisiones son el único comprador del trabajo.

11.1. Comparación con otros tipos de subnet

SN97 representa una competición por modelos: los miners publican students y los validators eligen un King mediante una evaluación multi-eje. SN53 representa una actividad diferente: inferencia de modelos abiertos con verificación de que el modelo declarado es el que ejecutó la request. La subnet propuesta se alinea inicialmente con el primer patrón y deja el serving para una fase posterior. [Ref. 1] [Ref. 2]

11.2. La afirmación “model factory”

Una arena que elige el mejor modelo para un challenge no demuestra todavía una fábrica generalizable. La afirmación se vuelve sólida cuando el protocolo evalúa pipelines de entrenamiento sobre workloads nuevos o parcialmente ocultos. Esa transición se refleja explícitamente en el roadmap.

12. Arquitectura conceptual y trust model

12.1. Componentes

Componente	Responsabilidad
Challenge registry	Versiona TaskSpec, reglas, base allowlist, hardware y deadlines.
Commit layer	Ancla hotkey, revisión, hashes, licencia y metadata del artefacto.
Artifact store	Aloja weights, adapters, containers y manifests inmutables.
Evaluation orchestrator	Programa pods, inputs ocultos, seeds y batches emparejados.
Metric workers	Ejecutan OCR, embeddings, optical flow, VLMs y checks específicos.
Human audit layer	Calibra aceptación y revisa dethronements o baja confianza.
Score engine	Aplica gates, normalización, coste y reglas de champion.
Public registry	Publica resultados, lineage, restricciones y demos reproducibles.

12.2. Trust model inicial

La evaluación de vídeo es cara y puede comenzar con un evaluator canónico operado en una infraestructura controlada. Para reducir confianza, el protocolo debe publicar código, schemas, seeds revelados después del commit, hashes, logs agregados y resultados firmados. Validators ligeros pueden verificar la firma y aplicar pesos mientras se replica gradualmente la evaluación pesada.

La evolución deseable es pasar de un evaluator canónico auditable a varios evaluadores pesados con batches emparejados y mecanismos de disputa. No es necesario fingir descentralización completa desde V0; sí es necesario definir cómo disminuirá la confianza con cada versión.

12.3. Reproducibilidad

- Hardware profiles fijos y publicados.
- Containers sin red y con dependencias pinneadas.
- Seeds, prompts, inputs y versiones revelados tras cerrar la evaluación.
- Cálculo de coste y latencia en el validator, no autodeclarado.
- Repetición periódica del champion para detectar drift o artefactos removidos.

13. Licencias, datos y cumplimiento

13.1. Open source, open weights y permiso comercial no son equivalentes

La subnet necesita una allowlist por versión. Una licencia permisiva como Apache 2.0 facilita derivados, pero muchas licencias community definen de forma amplia los modelos derivados, imponen thresholds comerciales, restricciones de transferencia o condiciones downstream. LTX-2 es un ejemplo de licencia que incluye expresamente fine-tunes, adapters y distillation dentro de la definición de derivados. [Ref. 6]

13.2. Teachers cerrados

No debe asumirse que poseer el output autoriza utilizarlo para entrenar un competidor. Los términos empresariales vigentes de OpenAI restringen, salvo excepciones, el uso de outputs para desarrollar modelos que compitan con sus productos. Los términos de Google Cloud restringen el uso de servicios u outputs para crear productos similares o modelos parecidos. BytePlus prohíbe expresamente utilizar sus servicios de vídeo, inputs, outputs o datos sintetizados para desarrollar, entrenar, afinar u optimizar otros modelos, salvo acuerdo específico. [Ref. 9] [Ref. 10] [Ref. 11]

REGLA DE PROTOCOLO

Los modelos cerrados pueden utilizarse como referencias de benchmark cuando sus términos lo permitan. No pueden convertirse en teachers de training por defecto. Cualquier excepción requiere autorización contractual explícita y documentada.

13.3. ProvenanceManifest

Cada submission debe declarar, como mínimo:

- Modelo base, revisión, licencia y territorio aplicable.
- Teachers y métodos de obtención de targets, logits, features u outputs.
- Datasets, origen, derechos de training, restricciones y retención.
- Uso de personas, voces, marcas, diseños, música o propiedad intelectual.
- Licencia del resultado y obligaciones para clientes o redistribuidores.
- Watermarks, metadata y obligaciones de transparencia preservadas.

13.4. Datos del primer challenge

Para reducir riesgo, el primer dataset debería priorizar renders 3D propios, identidades sintéticas, marcas ficticias y assets de un design partner con autorización expresa. Se deberían evitar vídeos extraídos de redes sociales, anuncios, celebridades, voces y música sin derechos claros.

13.5. Contenido sintético en la Unión Europea

El artículo 50 del AI Act aplica desde el 2 de agosto de 2026 y establece obligaciones de marcado legible por máquina para determinados contenidos sintéticos generados o manipulados. El serving futuro debe preservar provenance metadata y definir responsabilidades de provider y deployer. La arquitectura del registry debe registrar modelo, versión y origen del output desde el principio. [Ref. 12]

14. Primer challenge recomendado

14.1. Product Arc-8

Campo	Propuesta
Modalidad	Image-to-video.
Input	Imagen 3/4 de un producto; máscara o alpha opcional; prompt estructurado.
Transformación	Arco horizontal de cámara aproximado de ±8 grados; sujeto fijo.
Output	4 segundos, 1280×720, 24 fps, sin audio.
Restricciones	Preservar forma, color, texto, logotipo, fondo y número de objetos.
Hardware target	Una GPU de 24 GB para el track accesible.
Baseline	Wan 2.2 TI2V-5B y una solución 2.5D/depth-warping.
Métrica primaria	Coste total por clip aceptado tras superar gates de preservación y movimiento.

14.2. Por qué este challenge

- Se entiende visualmente en pocos segundos y produce material de lanzamiento claro.
- Image-to-video limita la entropía y mejora la posibilidad de medir identidad.
- Un arco pequeño evita exigir al modelo que invente la parte posterior completa del producto.
- El dataset puede generarse con 3D y ground truth de cámara, depth, máscaras y optical flow.
- Tiene una conexión comercial directa con ecommerce, agencias y creative automation.

14.3. Benchmark del challenge

Gate	Ejemplo de criterio
Identidad	Similitud visual y segmentación por encima del umbral.
Texto y logotipo	OCR y edit distance dentro de tolerancia.
Movimiento	Trayectoria y amplitud dentro del rango esperado.
Temporalidad	Flicker, drift y background change bajo umbral.
Aceptación	Panel humano considera el clip utilizable sin regeneración.
Eficiencia	Latencia, peak VRAM y coste medidos en el mismo hardware.

TEST DE REALIDAD

El challenge debe compararse contra un pipeline 2.5D o 3D. Si la alternativa determinista es mucho más barata y fiable, el workload puede ser una buena demo, pero no una buena oportunidad para un modelo generativo. La subnet debe buscar valor, no defender una técnica concreta.

15. Roadmap: de Model Arena a Model Factory

Fase	Qué compite	Qué demuestra	Qué no demuestra todavía
V0. Prueba interna	Equipo interno contra baseline.	Capability overlap y primer ahorro.	Valor de la descentralización.
V1. Genesis Model Arena	Modelos para un TaskSpec fijo.	Que miners externos pueden mejorar el champion.	Pipeline generalizable.
V2. Multi-Skill Registry	Champions por varios workloads.	Catálogo y repetibilidad de challenges.	Creación automática ante tarea nueva.
V3. Pipeline Arena	Recetas ejecutables bajo budget sobre task oculto.	La fábrica de modelos como capacidad reusable.	Serving descentralizado.
V4. Routing y serving	Modelos y proveedores de inferencia.	Producto, tráfico, feedback y economía externa.	Foundation model universal.

15.1. Claims permitidos por fase

En V1 el mensaje preciso es “competitive specialization arena for video models”. En V2 puede hablarse de un registry de especialistas. Solo en V3 resulta defendible presentar la subnet como una model factory, porque la unidad evaluada pasa del checkpoint a la capacidad de producir un nuevo checkpoint bajo condiciones no conocidas por adelantado.

15.2. Serving

La inferencia puede centralizarse inicialmente para controlar batching, cold starts, margen, UX y SLA. Se descentralizará únicamente si la red puede aportar disponibilidad, coste o resistencia a censura que supere la complejidad adicional. Entrenamiento y serving no deben unirse por inercia.

16. Criterios de éxito, riesgos y decisiones previas al lanzamiento

16.1. Gates de go/no-go

Gate	Pregunta de decisión
Capacidad base	¿El modelo permitido consigue una tasa de éxito no trivial en el workload?
Especialización	¿El fine-tune mejora claramente la aceptación en inputs no vistos?
Compresión	¿Reduce coste o latencia sin destruir la mejora de calidad?
Competencia	¿Se acerca al closed reference o lo supera en la restricción crítica?
Alternativa	¿Es mejor que una plantilla, 2.5D o 3D cuando estas son aplicables?
Demanda	¿Existe volumen y voluntad de pago suficientes para amortizar el specialist?
Descentralización	¿Los miners externos encuentran mejoras que el equipo interno no encontró?
Legalidad	¿El champion puede distribuirse o servirse en los territorios objetivo?

16.2. Riesgos principales y mitigación

Riesgo	Impacto	Mitigación
Calidad abierta insuficiente	El specialist nunca alcanza el umbral comercial.	Elegir task más estrecho, teacher abierto más capaz o abortar el nicho.
Cold start tecnológico	Los primeros miners solo replican LoRA básica.	Baseline fuerte, grants de investigación y libertad de receta.
Goodhart y reward hacking	Submissions optimizan OCR/VLM sin utilidad humana.	Gates, métricas múltiples, human audits y test dinámico.
Overfitting	Gran score público, mala generalización.	Hidden assets, procedural variation y canaries.
Proxying	Miner llama a una API cerrada.	No-egress, fingerprinting, reproducibilidad y latency tests.
Licencia incompatible	Champion no explotable.	Allowlist y ProvenanceManifest antes de GPU eval.
Poco volumen por cluster	No se amortiza training.	Opportunity score y design partners antes del challenge.
Base models evolucionan rápido	Vida útil corta del specialist.	Adapters portables, challenge rounds y amortización conservadora.
Evaluación cara	El validator consume emisiones sin crear valor.	Prechecks CPU, sampling adaptativo, paired eval y challenger gates.

16.3. Decisiones que deben cerrarse antes de mainnet

- TaskSpec exacto del Genesis Challenge y criterios descalificantes.
- Modelo base, revisión, licencia y hardware profile.
- Propiedad y licencia de los champions públicos y privados.
- Proporción de evaluación automática y humana.
- Regla de rewards, challenger rewards y dethronement margin.
- Trust model inicial: evaluator canónico, validators ligeros y auditoría pública.
- Budget máximo de entrenamiento y de evaluación por submission.
- Design partner, volumen y baseline económico externo.

16.4. Evidencia mínima para salir

- Un baseline reproducible y legalmente aprobado.
- Un challenger interno que lo supere en held-out.
- Un evaluator que produzca resultados estables al repetirlo.
- Una comparación honesta contra closed reference y alternativa determinista.
- Un leaderboard conectado a submissions reales.
- Un dataset con provenance y un hidden set protegido.
- Al menos un design partner o fuente verificable de demanda.

CONCLUSIÓN

La razón de existir de la subnet es coordinar I+D competitiva para convertir workloads repetitivos de vídeo en modelos especialistas con valor económico verificable. El primer objetivo no es construir un producto generalista ni descentralizar inferencia; es demostrar que las emisiones producen un modelo mejor, reutilizable, legal y más eficiente que las alternativas para una tarea concreta.

Anexo A. Esquema mínimo de TaskSpec

El TaskSpec es el contrato canónico del challenge. Debe ser suficientemente preciso para que miners y validators implementen la misma tarea sin coordinarse fuera del protocolo.

Campo	Descripción
task_id / version	Identificador único e inmutable de la versión.
modality	T2V, I2V, V2V, audio-to-video u otra modalidad.
input_contract	Tipos, dimensiones, máscaras, referencias y metadata permitida.
output_contract	Resolución, fps, duración, codec, audio y aspect ratio.
semantic_distribution	Familia de prompts, categorías, materiales y transformaciones.
hard_constraints	Condiciones cuya violación descalifica el output o modelo.
quality_gates	Umbrales de identidad, temporalidad, adherencia y aceptación.
hardware_profile	GPU, VRAM, CPU, memoria, precisión y timeout.
training_budget	GPU-hours, datos permitidos y deadline.
model_allowlist	Bases, revisions y licencias autorizadas.
teacher_policy	Teachers permitidos, prohibidos o sujetos a contrato.
evaluation_policy	Seeds, número de samples, human audit y dethronement.
artifact_license	Reglas de apertura, gating, uso privado y downstream.

Anexo B. Esquema mínimo de SubmissionManifest

Grupo	Campos mínimos
Identidad	hotkey, challenge, commit block, artifact revision, hashes.
Modelo	base, arquitectura, parámetros, precision, adapter type, sampler, pasos.
Ejecución	container digest, command, environment, hardware, VRAM estimada.
Training	método, budget, optimizer, dataset manifests, teacher lineage.
Legal	licencias, territorios, restricciones comerciales, transfer y AUP.
Seguridad	safety filters, watermarks, prohibited uses y model card.
Disponibilidad	storage URI, retention period, mirror y contact.

Anexo C. Glosario

Término	Definición
Workload	Distribución recurrente de inputs, prompts, constraints y outputs esperados.

Término	Definición
Challenge	Competición versionada que operacionaliza un workload mediante un TaskSpec.
Teacher	Modelo o sistema cuyas señales ayudan a entrenar al student.
Student	Modelo candidato optimizado para el challenge.
Specialist	Modelo con cobertura estrecha y rendimiento alto dentro de un contrato.
Champion	Candidato líder para una versión de challenge y hardware profile.
Registry	Catálogo de artefactos, lineage, licencias, scores y perfiles de coste.
Distillation	Transferencia o compresión de comportamiento, pasos, features o distribución.
Coste por aceptado	Coste esperado de obtener un output utilizable, incluidos reintentos y amortización.
Pareto frontier	Conjunto de modelos no dominados simultáneamente en calidad, coste y latencia.
No-egress	Entorno de evaluación sin acceso a redes externas.

Referencias

1. Distil — SN97: competitive model distillation on Bittensor. https://github.com/unarbos/distil
2. Engy — SN53: verified inference for frontier open models. https://github.com/hanlinai/engy
3. Wan 2.2 — repositorio, modelos, requisitos y licencia Apache 2.0. https://github.com/Wan-Video/Wan2.2
4. HunyuanVideo 1.5 — modelo, training y step-distilled checkpoints. https://github.com/Tencent-Hunyuan/HunyuanVideo-1.5
5. NVIDIA Cosmos3-Super-Image2Video-4Step — model card. https://huggingface.co/nvidia/Cosmos3-Super-Image2Video-4Step
6. LTX-2 — Community License Agreement. https://github.com/Lightricks/LTX-2/blob/main/LICENSE
7. VBench: Comprehensive Benchmark Suite for Video Generative Models. https://arxiv.org/abs/2311.17982
8. VBench-2.0: Advancing Video Generation Benchmark Suite for Intrinsic Faithfulness. https://arxiv.org/abs/2503.21755
9. OpenAI Services Agreement, restricciones sobre uso de outputs para modelos competidores. https://openai.com/policies/services-agreement/
10. Google Cloud Service Specific Terms, AI/ML competitive use and model restrictions. https://cloud.google.com/legal/archive/terms/service-terms/index-20260504
11. BytePlus Specific Terms for Video Generation Model Services, incluyendo Seedance. https://docs.byteplus.com/en/docs/modelark/Specific_Terms_for_the_BytePlus_Video_Generation_Model_Services
12. Comisión Europea: obligaciones de transparencia del artículo 50 del AI Act. https://digital-strategy.ec.europa.eu/en/faqs/transparency-obligations-under-article-50-ai-act

Acceso y revisión de fuentes: 26 de agosto de 2026. Las licencias y términos de servicio pueden cambiar; deben validarse de nuevo antes de cada challenge o despliegue.

FIN DEL DOCUMENTO