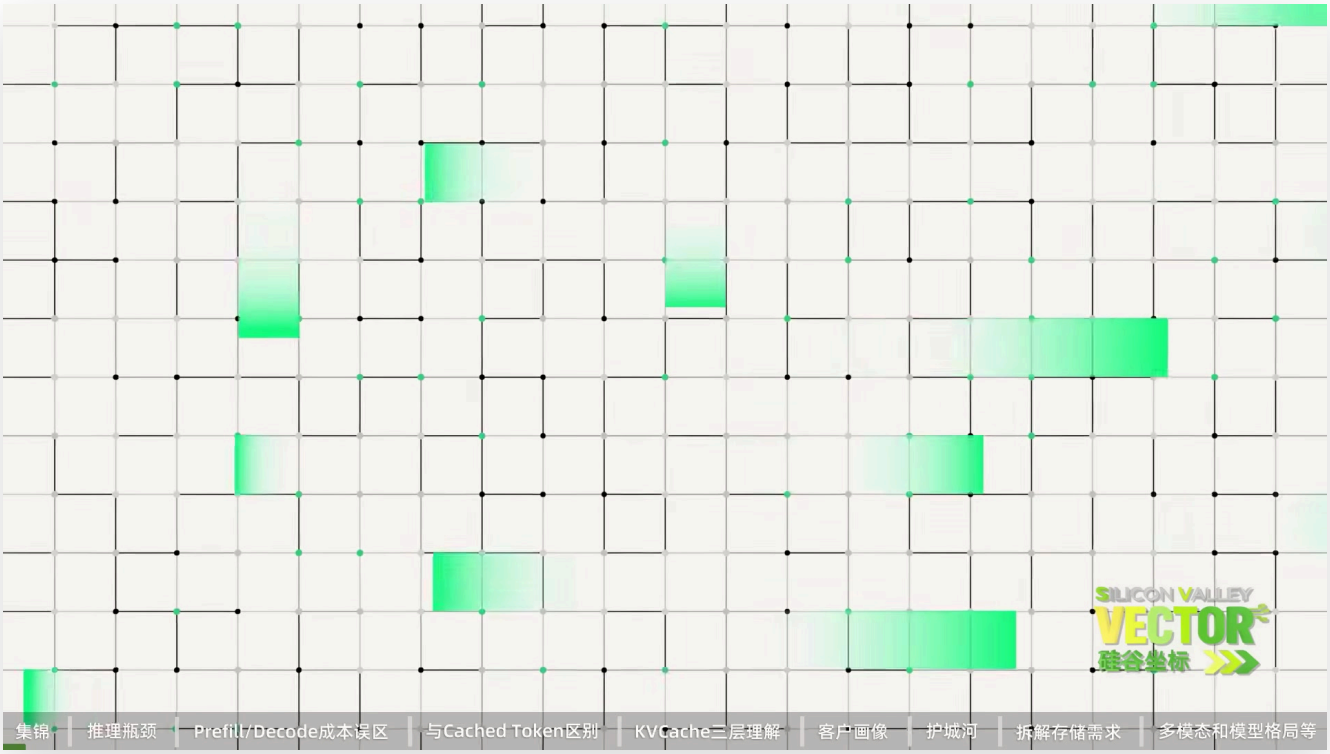
硅谷坐标 x Tensormesh 江鋆晨:AI 的记忆–KvCache的三层理解 – 图文解说

> 一张关键画面对应一段完整解说，按原视频时间推进。

文章摘要

本期《硅谷坐标》与 Tensormesh 的江鋆晨从三层视角解释 **KV Cache**：从可缓存的**黑盒数据**，到可被理解和编辑的语义**白盒**，再到面向 **Agent**、**推理**和**长期记忆**的系统层。对谈进一步讨论 **Prefill/Decode** 成本、**存储**层、模型状态、**KV Cache** 的**商业化**时点，以及 AI 规模化后**内存**与数据中心的关系。**核心判断是：KV Cache 不只是临时缓存，而可能成为连接模型推理、存储和下一代 AI 基础设施的关键层。**

01. KV Cache 的三层理解（00:02–00:40）



场景 1

本节主旨 KV Cache 的三层理解

完整内容

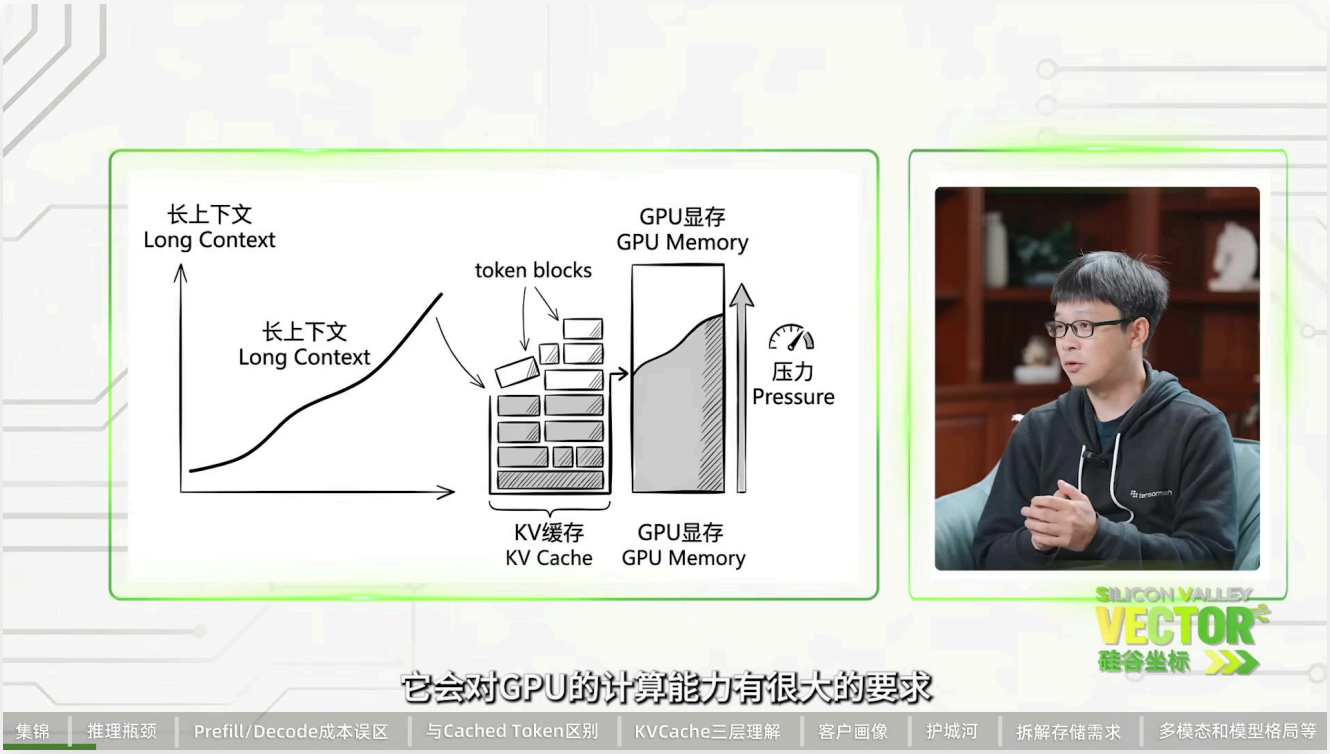
江鎏晨：KV Cache 有**三层理解**。第一层理解是，**KV Cache** 是一种可以**存储**下来的**黑盒数据**，很多工业界的公司都在做。第二层理解是，工业界现在做得很少，但是学术界已经做了很多：KV Cache 是有**语义信息**的，可以把 KV Cache 看成一个**白盒**，你可以去改动它的内容。第三层来说，你可以继续改它的语义，学术界已经开始有人做了，但是工业界很少有人理解到这一层。

我们为什么做这个公司，原因之一是在学术界里面，可能大家已经走得很远，但是你需要影响工业界的话，你必须得做一个这样的大的 startup。这个 KV Cache moment 可能比我想象的要来得早，有可能在今年年底前就会有很多。

画面说明

画面是《Silicon Valley Vector / 硅谷坐标》的片头图形：白底网格上分布着黑色、灰色和绿色节点及矩形块；右下角有节目标识。底部导航栏可见“集锦”“**推理**瓶颈”“**Prefill/Decode** 成本误区”“与 Cached Token 区别”“**KVCache 三层理解**”“客户画像”“护城河”“拆解**存储**需求”“多模态和模型格局等”等文字。

02. AI 推理的资源瓶颈（00:54–02:09）



场景 2

本节主旨 AI 推理的资源瓶颈

完整内容

曹卿云：江鋆晨，欢迎你来到《硅谷坐标》。今天在谈 TensorMesh 之前，我想请你帮我们建立一个认知的坐标。在现在整个 AI **推理**的环节里，主要的瓶颈在哪里，卡在哪些环节?

江鋆晨：首先，AI 已经不是两年前的 AI 了。它的 workflow 比以前长了很多很多，但同时，这个 workflow 当中的 prompt pattern 和以前也非常不一样。以前可能大家就是说一些对话性质的东西，现在 use case 多了，有很多 agent use case 会有长文本。**同一个 query，它的文本一长、输入越长，就会对 GPU 的计算能力有很大的要求。**

同时，它会对 GPU 内部的**存储**有非常大的要求，因为你算得越多，它存的数据也需要越多。同时它有 model 的记忆，因为 model 如果把长文本梳理完了之后，会有很多内部的记忆，这个记忆以后也得存下来。**所以现在的瓶颈说到底就是资源跟 workflow 可能有不对等。**不仅是通过改模型、等待更好的 GPU，同时也需要更好的软件层面的创新，让这么大的 workload 在有限的资源上可以跑得好。

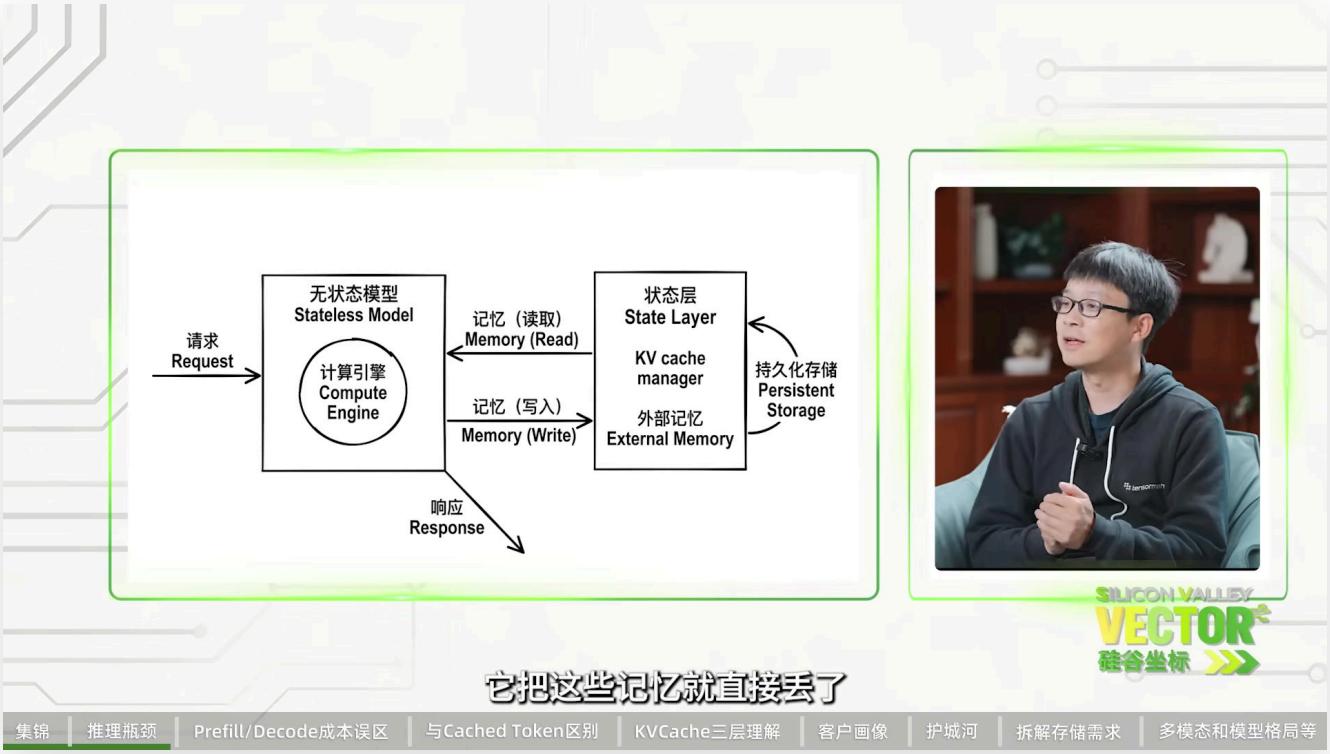
曹卿云：那 TensorMesh 在这里面是解决哪一个环节的瓶颈呢?

江鋆晨：TensorMesh 是一家做软件创新的 startup。

画面说明

左侧是“长上下文 / Long Context”曲线，指向 token blocks，再指向标有“GPU 显存 / GPU Memory”的柱形图和“压力 / Pressure”图标；右侧是江鋆晨的访谈画面。**底部字幕写着“它会对 GPU 的计算能力有很大的要求”。**

03. 模型记忆与 KV Cache 复用（02:10–03:31）



场景 3

本节主旨 模型记忆与 KV Cache 复用

完整内容

江鋆晨：在软件这个层面呢，我们做的主要就是大模型的记忆管理。为什么说是记忆管理呢？其实大模型每次看到很多文本之后，它自己可以有一个内部的理解，管它叫 **KV Cache**。**KV Cache** 其实是一大堆浮点数，这些 tensors **存储**了模型自己的记忆，模型自己对于长文本或者任何 token 的理解。

这个记忆在 Transformer 做这些软件之前，经常会被 GPU 丢掉，被模型丢掉，因为模型一旦看过一次、做过一次输出之后，它是 stateless 的，它不会记这些记忆，它把这些记忆就直接丢了。**但是 TensorMesh 和我们做的开源项目可以把这些记忆留存下来。**然后任何时候 model 如果再看到类似的场景、类似的 input、同样的 context，模型可以让存下来的记忆复用起来。

这件事情呢，TensorMesh 现在估计是做得最好的几家公司之一。

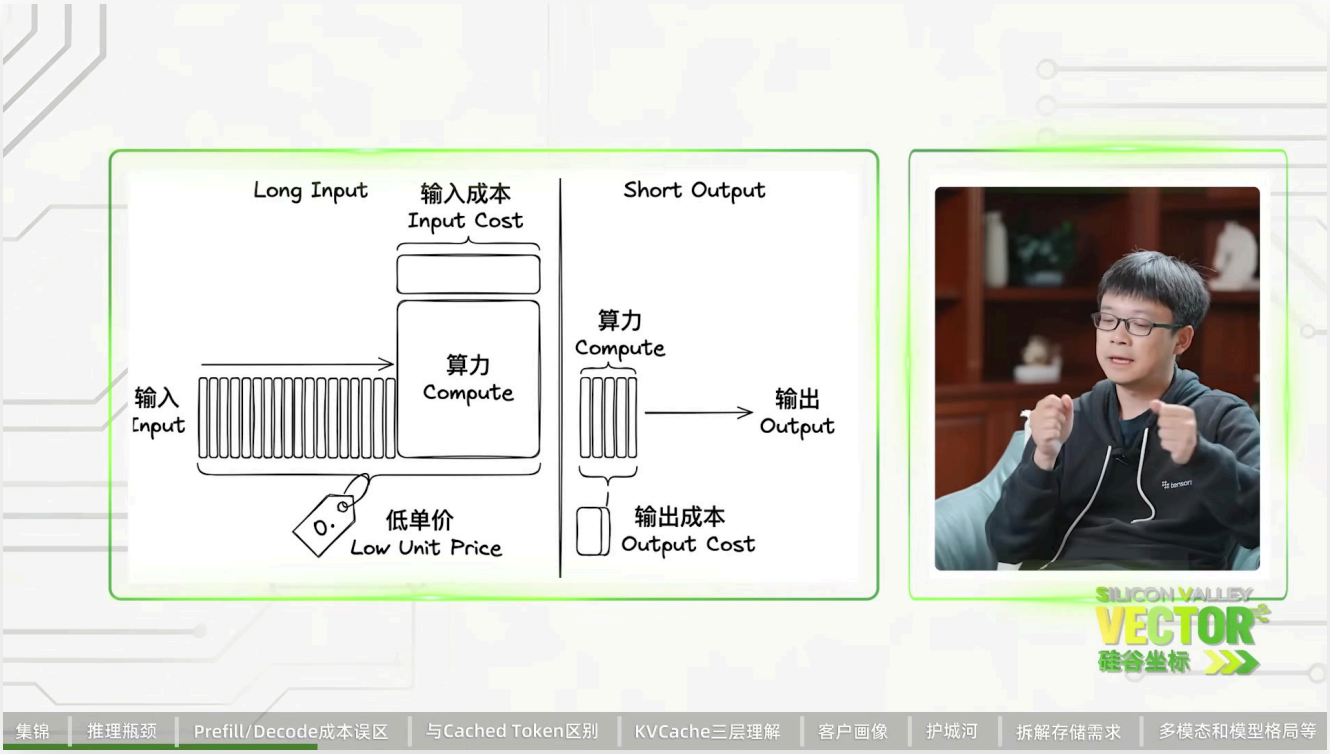
曹卿云：同时呢，我们的技术也有不少领先，所以我觉得我们在解决现在最卡脖子的难题之一。所以我理解就是说，在模型输入的 prefill、预解读阶段，和 decode、output 的阶段，然后你们解决的主要是在 prefill 这个阶段发生的事情，对吧?

江鋆晨：对，对，对。

画面说明

画面显示一个“无状态模型 / Stateless Model”与“状态层 / State Layer”的架构图：左侧有“计算引擎 / Compute Engine”，右侧列出“KV cache manager”和“外部记忆 / External Memory”；两侧用“记忆（读取） / Memory (Read)”“记忆（写入） / Memory (Write)”箭头相连，右侧还有“持久化**存储** / Persistent Storage”，左侧标有 Request，底部标有 Response。右侧为江鋆晨访谈画面。

04. Prefill 成本与 Output 成本（03:32–07:21）



场景 4

本节主旨 Prefill 成本与 Output 成本

完整内容

江鋆晨：暂时是。**Decoding 也有一定的用途，如果你的输出太长，GPU memory 也存不下了，那所以在输出期间也有可能需要一些远端的设备来帮你维护，或者一个分开的设备来帮你存 KV Cache。**但是我们主要做的，还是帮你省去输入不停地读同样文本时的 overhead。

曹卿云：讲到这一点，大家有一个感觉，就是输入端的 input token 价格很便宜，是 output 的可能只有几分之一。**听起来好像不是一个能 create 很多 value、创造很多价值的创业方向。**

江鋆晨：**其实你如果看真正的计算成本，不去看它的定价，而去看背后的计算成本，你处理一个 input token 和生成一个 output token，它们的代价是差不多的。**并不是说处理一百个 token 的 input、生成一百个 output tokens，input 就便宜很多，它其实计算的量是差不多的。

为什么会有这样一个印象，说 input 比 output 便宜呢？因为大家在用 Chatbot 这种 application，或者说用 LM 的时候，多数时候是在等 output，多数时候花时间在看 output 慢慢生成出来。大家会下意识忽略 input，好像过了几秒钟就直接开始生成 output 了。**但其实在预处理、prefill input 的期间，有很多计算需要做，甚至它会做比 output 的那 1000 个 token 还要多的计算。**很多时候模型在处理 input 的时候，可以大规模并行化，但并行化不代表计算量很小，它其实做了很多计算。

你如果算这个 FLOPS，GPU 的计算单元的话，其实 input 和 output 在每个 token 上的计算成本是差不多的。甚至在很多真实的 use case 下，input 的长度比 output 长。比如 input 几万个 tokens，output 可能几千个 tokens，input 确实比 output 长十几倍。但是如果看时间，用户盯着屏幕的时候，有多少时间是在等 input 被处理，有多少时间是在等 output 生成？Input 其实占的时间可能只有十分之一，**90% 的**时间都在 output 上。

所以大家会有一个印象：output 非常贵，是因为它比较慢，provider 需要花很多精力把 output 做出来。但其实 output 只是处理的线性化，它没有并行而已。**因为它线性化、没有并行，所以占用 GPU 的时间比较多。**从 provider 的角度来说，它确实会看起来更贵一点，也有更好的理由去说它很贵。但如果算真正的成本，prefill 的成本跟 output 的成本在每个 token 上是一样的，甚至整体的 prefill 成本会更高。

曹卿云：**所以 prefill 的 input token 价格便宜，但它的数量级其实是 output 的好几倍。**

江鋆晨：对。因为模型是 stateless、没有状态的东西。如果是 agent application 或者长对话，over time，或者说你跟模型交互得多了，模型每次都得知道更多 context，**所以模型的 input 只会越来越长、越来越长，**但每次的 output 不会相应地变长。很多时候 agent application 和 chatbot application 的输入，过一段时间之后就会比输出长很多。

回到你刚刚说的，为什么大家看 pricing 的时候会有这个印象？**我觉得很大程度上是这些 inference provider 需要平衡成本。**它并不是说 input 便宜、output 贵，真的反映了它真实的 cost。

曹卿云：大模型的定价里面，除了 input token、output token，还有一个 cache token。**Cache token 相对于这两个来说是白菜价。**感觉大模型已经也在做同样的 cache，跟你们做的事情有什么不一样？

画面说明

左侧图示对比“Long Input / 输入”和“Short Output / 输出”：输入端有较长的 token 序列、较大的“算力 / Compute”区域和“输入成本 / Input Cost”，并标注“低单价 / Low Unit Price”；**输出端 token 序列较短、标注“输出成本 / Output Cost”。**右侧是江鋆晨访谈画面。

05. Prompt Caching 与三层理解（07:22–09:08）

场景 5

本节主旨 Prompt Caching 与三层理解

完整内容

江鋆晨：首先，prompt caching，就是你刚刚说的一些 token 被 cache 下来，然后在账单上也会体现成一个更低的价格。这是很多现在大模型**推理**服务商会提供的服务。但是在我们的开源项目和所有用我们开源项目的公司里面，做所谓 cache 下来的 token，它的应用场景就非常有限。

因为第一，它 cache token 只是在 CPU 里面存下来。你要把它放到 remote storage、放到更便宜的**存储**系统里面，性能会差很多。这也是我们经常观察到的事情。我们公司的目标不仅是优化 KV Cache **存储**的效率，同时还要做一套可以让所有人用，并且可以有很多大家现在根本想不到的 KV Cache 相关优化。

在我们看起来，KV Cache 是未来大数据。为什么这么说呢？KV Cache 本身并不是把它存下来以后只能复用的东西，它里面有有效的信息、非常有用的**语义信息**。这些**语义信息**，是现在所有把它 cache 下来的人不会去看的。把这些语义看出来一些有用的东西，它可以让你把 KV Cache 存得更多、存得更好、存得更快，甚至可以改变 model 的行为。

这些大模型**推理**服务商在 cache token 这件事情上，确实是一个非常好的 validation，说明我们做的东西确实有价值。他们现在提供这个服务，只不过是我们所有 vision 里的一开始这一小步。

曹卿云：所以你们的 vision 是什么？你对于 KV Cache 的本质是怎么理解的？

江鋆晨：我们看 KV Cache 呢，有**三层理解**。第一层理解是，KV Cache 是一种可以存储下来的**黑盒数据**，是一个 compute state。这个状态可以被存下来。

画面说明

画面是曹卿云的近景访谈镜头，底部字幕写着“在我们看起来 kvcache 是未来大数据”，右下角有《Silicon Valley Vector / 硅谷坐标》标识；底部导航栏仍显示 **KVCache 三层理解**等栏目。

06. KV Cache 的黑盒、白盒与语义层（09:08–11:43）

场景 6

本节主旨 KV Cache 的黑盒、白盒与语义层

完整内容

江鋆晨：在第一层理解里面，大家觉得这个状态就是个黑盒，你不能改。因为你存下来之后，就只能 as is 存什么用什么。**这个层面其实有很多公司在做，很多工业界的公司都在做，因为这是最基本的一层理解：你把它记下来以后可以复用。**

第二层理解呢？是工业界现在做得很少，但是学术界已经做了很多。是什么呢？是 **KV Cache 有语义信息**，代表模型自己的注意力信息、attention。**所以如果把 KV Cache 的值稍微改一改，只要不改变它的注意力、不改变它的语义信息，这个 KV Cache 一样可以复用，而且复用可以很好。**

到这层理解，大家就不会再把 KV Cache 看成黑盒，而会把 KV Cache 看成**白盒**。你可以改动它的内容，可以让它变得小一点，可以做 lossy compression；或者你可以把 KV Cache 做一些 update，使得它的文本复用在任何地方都可以复用。甚至，KV Cache 可以在文本不同的时候，对应的文本不同，但可能语义差不多，那这样的话还可以复用它的 KV Cache。我们有些技术可以让 KV Cache 在不同的模型之间复用。

这些都是到第二层理解：理解到 KV Cache 不仅是黑盒，它是一个有语义信息的东西。你改变它的数字，只要不改变它的语义，就可以做很多优化。**这些学术界做了很多，但是工业界现在很少做。**

第**三层理解**，我觉得是最有意思的理解。大家想象一下，其实模型在生成结果的时候，取决于两个东西：要么取决于模型的 weight、权重；另外一方面，它其实取决于 KV Cache。你可以想象，模型的输出就是模型的权重和模型生成的 KV Cache 当中做一些操作，生成它的结果。

所以只要你可以把 KV Cache 做一些改变，就可以改变模型的输出，可以改变模型的行为。**第二层只不过是可以改 KV Cache，但是不改它的语义；第三层的时候，你可以继续改它的语义。**你可以告诉模型，“pay more attention to this, pay less attention to that”，可以让它去改它的注意力。做这种操作，甚至可以让模型的输出准确性更高、质量更高。

这些事情学术界已经开始有人做了，但是工业界很少有人理解到这一层。**那我们为什么做这个公司，原因之一就在学术界里面，可能大家已经走得很远，但是你需要影响工业界的话，你必须得作为这样的 startup。**

曹卿云：我就很好奇，感觉 KV Cache 应该是工业界每天都在接触的东西，但是最后这块的创新、前沿东西却是学术界在做，这是为什么呢?

画面说明

画面是主持人与嘉宾的双人访谈全景，底部字幕写着“这些学术界做了很多”；两人分别坐在画面左右的椅子上，中间有小桌、瓶子和杯子，右下角有节目标识。

07. 跨学科打通 KV Cache 三层（11:43–15:18）

场景 7

本节主旨 跨学科打通 KV Cache 三层

完整内容

江鋆晨：在工业界里面看，工程师看到系统里面产生的数字时，第一反应是不去改它，而是把它存起来，然后把它当黑盒看待。学术界看待这种数据问题，会想：这些数据里面有什么**语义信息**？类似的事情在当年的大数据时代也发生过。

现在 AI 时代也会发生。**AI 时代里面最重要、最有意思的数据就是 KV Cache。**我们公司一直说，KV Cache 是未来的大数据。其实就是这个数据本身有价值，而不是把这个数据存下来、复用一下的价值。

曹卿云：**Beyond that，你觉得工业界现在只是理解到第一层，第二层和第三层对他们来说，如果他们也想做的话，你们的优势在哪里？**工业界很快就能有更多资源、更多人力把这个事情做掉了。

江鋆晨：完全有可能，而且我甚至觉得这件事情本来可能明年、后年再发生，但是现在有可能今年年底就会发生。KV Cache 的关注度越来越高，自然而然就会有人说：我能拿 KV Cache 做什么更有意思的事情?

但是我一直有信心说，这件事情其实对做 startup 来说是非常好的机遇。为什么呢？你要对 KV Cache 这三层都有理解，是非常非常难的事情。能做第一层、做 KV Cache 黑盒操作的人，必须得是非常好的 engineer，必须得是 infra engineer、backend engineer，能接触到 GPU 的 engineer，才能看得到这个数据。

但是这些人能做第一层，做不了第二层，因为他不知道这个数据到底有什么含义。你必须得是做 ML 的人，才能理解这个数据的含义，才能做第二层，甚至第三层。但是做第二层和第三层的人是谁？是 researchers。这些 researchers 并没有工程的 inside，不能让他接触到真的 KV Cache system。所以能做第二层、第三层 research 的人，做不了第二层、第三层的 engineering，因为他必须得有第一层 engineering 的实力。

精简一点，其实就是做这些不同层的人的背景不一样。**在大公司里面，可能一个公司有做这三层的人，但很难把他们放在一起去做一整套系统。**

曹卿云：可以把这三层全打通。**所以看起来你们的优势是跨学科、跨专业。**跟大厂比起来，在 CS 这个领域里面，可能是要跨 topics。

江鋆晨：所以大厂组织这些资源会更困难。一方面是他们现在还没有这个 vision，没有这种魄力去真的做这件事情。有很多公司内部要做类似的东西时，他们会说：TensorMesh 融了那么多钱，就干这个，是吧？我们也应该做这个事情。

其实我并不是特别担心这件事让大公司知道了，他们会立刻 act that fast。**你听没听 Sam Altman 说过这个事？**他以前有个观点，就是小公司做什么，你有再好的 vision，去大公司 CEO 面前跟他说“你应该做这个”，他们都不一定做得了。不是因为他没有这个实力，是因为人的 attention 分散的地方太多了。

我们是专注做这一件事情。大公司有很多 priorities，他们可能有一个组在做这个，但是可能过两个月就做别的了。**所以我们现在有个机会：我们是非常少数的 startup 或者开源项目，直接只做这一层、只做这一个 vision。**但这个 vision 过两三年之后，会变成非常非常大的一件事情。

等大家都意识到第二层、第三层有多少优化空间的时候，到那时候 KV Cache 就会真的变成 big data。你想想看，当年 big data 有多少多少公司，对吧?

曹卿云：讲到职场空间，我很好奇，你们解决的这个问题，都是一些 query 可能更多是重复调用的公司吗？都是什么样的客户，什么样的场景会是——

画面说明

画面是曹卿云的单人访谈近景，底部字幕写着“其实就是做这些不同层的人的背景不一样”；右下角有《Silicon Valley Vector / 硅谷坐标》标识，底部导航栏可见“**KVCache 三层理解**”等栏目。

08. 共享知识、Agent 与 KV Cache（15:18–20:53）

场景 8

本节主旨 共享知识、Agent 与 KV Cache

完整内容

曹卿云：你们最好的 customer 是什么样的?

江鋆晨：这是好问题。其实首先，我们不能看到客户到底在用什么、这个 library 在用什么。甚至开源项目，我们都不知道谁在用这个东西，只知道用的人很多很多。**因为很多公司，我们上去问他说：“你们要不要试试看 LMCache?”**他说：“哦，我们一直在用啊。”

所以确实很难知道这家公司到底 exactly 怎么在用。但是从我们和他们交流的经验来看，大企业有很多情况，包括 coding agent、公司内部的 enterprise chatbot，包括内部的 financial services、legal services。它有很多所谓 shared knowledge：公司内部可能有一个 code base，需要被所有

agent、所有 coding agent 使用；有一堆 policy documents，需要给所有 chatbot 使用；有一些 legal documents，需要给所有 legal chatbot 使用。

有很多这种 shared knowledge 需要在不同 use case 里面使用。但这个时候，你要每次让 LM 不停地读吗？这肯定很浪费。**所以现在一种非常 common use case，就是 across multiple applications，包括 coding、chatbot、RAG、multimodal 这些应用。**Across many applications，它都有 shared knowledge 作为模型输入的 context。这个场景是对 LM Cache 和 tensor cache 技术最有利的场景。

曹卿云：我可以理解成，企业端有一些私有部署的需求，希望能够把控自己的 **KV Cache**，而且 workflow 有很多重复调用，比如法律、金融。可能还有客户服务这种?

江鋆晨：对。有些 best practice，一个 agent 效果好不好，很取决于你怎么跟它描述这件事情。很多 best practice 确实可以以文本的形式存起来，以 human-readable 的方式存下来。**但是如果可以把它用模型、用 model-readable、model-native 的方式存下来，模型可以直接读，这样的话——**

曹卿云：**不用每次再处理了。**

江鋆晨：对。

曹卿云：这个很有意思。**所以你觉得这样的市场空间，agentic workflow 越来越多，其实对于你们更利好，因为 prefill 这种重复理解问题会更多?**

江鋆晨：会越来越多。因为 agent 说到底，是一个和环境以迭代方式交互的逻辑。你迭代越多，交互越多，产生的 history 就越多。**History 越多，每次对 model 来说就是 long context，longer and longer context。**

江鋆晨：**除了降本这个角度之外，比如这个 agent 因为有一个缓存，会不会整个 agent 最后的任务效果也变得更好?**

江鋆晨：会有，但这不是我们现在产品里面已经做的、现在 vision 里面一定会做的事情。**因为模型把记忆存到我们这里了，首先我们觉得需要明确一件事情：模型是怎么生成的?**

模型是把你给它的内容读一遍，读完一遍就生成一个 KV pair。它其实是 one-directional 的，就是读一遍。第一次生成出来的 KV pair，经常会有所谓“lost in the middle”的概念。**就是说，如果文本太长、input 太长，这个时候你让模型找出文本里面某些特定的具体信息，如果信息在开头和结**

尾，模型一般找得比较好；但是如果信息在中间，模型就找得不太好。

这其实跟人一样。人如果读一本长书，只读一遍，肯定会漏掉很多信息。我们的 vision 之一就是把 **KV Cache** 存下来，同时可以在**存储**的地方对它做优化。可以让模型把文本重要的地方再读一读，把 attention 再提高一点。这样的话，就可以把 KV Cache 的内容变得更好。下次这个 KV Cache memory 再被复用的时候，模型就可以更好、更精确地输出。

曹卿云：如果看整个你们服务的市场空间，total addressable market 会有多大，增速怎么样?

江鋆晨：我觉得 total addressable market 是个 misleading word，因为 addressable 的前提是大家现在想这个事情。**绝大多数人现在都没有在想这个事情，因为大家没有意识到有这个潜力。**

现在在企业里面做 AI 的人基本有两类：**要不然是 train model 的人，要不然是写 agent application 的人。**能提高一个 agent quality 或者提高 agent accuracy 的人，就这两类人，因为你大不了要不然改模型，要不然改 prompt。

在模型已经确定、prompt 也确定好的前提下，怎么让模型更好地理解这个 prompt？这件事情是所谓提高 quality 的第三条路。**这条路现在其实看的人并不多，谈的人很少，所以大家根本没有意识到有这样一个 market。**

能做这件事情的人，既要懂 system，也要懂 machine learning。因为你要打通从第一层**存储**一直到最后的优化。你需要一个既有 system 能力、又有非常强 system 能力，同时也有非常强 ML 能力的团队。说实话，这种人才是有的。我从学校里面出来，有很多 CS 毕业生，或者做 ML 的人，有很强的 system 能力。

曹卿云：或者做 system 出身，但有很强的 ML 能力。但这些学生一般去了工业界，现在都拿很高的工资去纯粹做 model training 了。**现在行业里好像做 KV Cache 优化的公司也很多，那 TensorMesh 跟其他公司有什么不一样?**

江鋆晨：我们做这个事情比所有人都早，我们是第一个开始做这方面 research 的，**做 KV Cache 系统优化的 research**；然后我们是第一个做开源的，同时现在也是最早推出 commercial product 的团队之一。

画面说明

画面是江鋆晨的单人访谈近景，左下角有问题条“缓存除了降本，还能提升效果吗”，底部字幕写着“因为你模型把记忆在我们这儿存起来了”；右下角有节目标识。

09. LMCache 的开源生态与系统能力（20:54–22:06）

场景 9

本节主旨 LMCache 的开源生态与系统能力

完整内容

江鋆晨：最开始有这种 commercial product 出来的。**然后我觉得，如果从开源生态角度上来说，我们 LMCache 这个项目，估计是类似 effort 里面，包括学术界、工业界，ecosystem support 最好的。**

当然，LMCache 这个项目现在已经不仅仅是我们公司在维护，有很多很多公司都在维护。你如果问我技术上，就系统技术上、做 **KV Cache 存储**上，我们做的东西跟别人确实没有太大区别。

但是你要说我有一套真的系统、真的 artifact，把这件事做好，并且让很多人都可以顺利地用起来，还有很多公司以开源的形式做贡献，我有信心说，没有其他公司可以做得那么好。

另外一方面，我跟别人不一样的地方是，绝大多数做 **KV Cache** 优化的公司，他们所谓的优化就是把它存下来，最多做做压缩，也就做到这个程度了。很难有公司有能力把系统优化、我刚才说的那些优化都打通。这种公司我们现在还没看到。

但是像我说的，可能这个 KV Cache moment 比我想象的要来得早，有可能在今年年底前就会有很多。说实话，我听过投资人问我：“这个公司在融钱，干的事好像跟你们差不多。”我说：“对，确实差不多。”

画面说明

画面是江鋆晨单人近景，底部字幕写着“并且让很多人都可以顺利的用起来”；右下角有《Silicon Valley Vector / 硅谷坐标》标识，底部导航栏可见“与 Cached Token 区别”和“**KVCache 三层理解**”。

10. CacheBlend：非前缀复用（22:07–23:41）

场景 10

本节主旨 CacheBlend：非前缀复用

完整内容

江鋆晨：Good luck。你刚刚说到其他优化，我觉得 compression 是一个比较特殊的优化。**Compression 是大家非常容易理解的，因为你说 video 也有 compression，各种各样的东西，你要大了，第一想到的就是 compression。**

但有很多只在 **KV Cache** 上 make sense 的优化。比如刚才说的 attention，你可能改这个 model attention。我给你举另外一个例子，叫 CacheBlend。CacheBlend 是我们估计独有的技术，是做什么呢？传统上 **KV Cache** 只能在前缀上做复用。如果要复用的 text 不在前缀上，而在一个 input 的中间，那 KV Cache 不能复用，因为 KV Cache 里面不包含和它当时那个前缀的关系。

这是一个非常技术性的问题：KV Cache 需要被 update 一下，需要被改。改一下之后才能复用。这个改的过程需要计算。**所以 KV Cache storage 不仅仅是 storage，它是一个 service，里面有 smart intelligence。**你要把里面存的内容改一下，才能让它复用起来，才能让 GPU 去复用。

这样的话，在这个 storage system 里面就需要有计算能力。所以它不仅仅是**存储**问题，也是计算问题，这里面牵扯到很多：你真的需要懂系统，也需要懂 ML。

曹卿云：好像听到的是这种对于前缀的修改，没有再听说过对于后缀的?

江鋆晨：就非前缀，你要中间也是，对吧? 也是非前缀。

曹卿云：对，非前缀。还有一个就是说，我们知道未来 context window 会继续增大，包括 agent 的需求也会增多。**特别是现在我们还只是讲 language model，之后还有 image 以及 video。**

画面说明

画面是曹卿云与江鋆晨的双人访谈全景；底部字幕写着“这个改的过程呢”，右下角有节目标识，画面中可见两把椅子、中间的小桌和背景书架。

11. 多模态输入带来的 KV Cache 规模（23:42–25:15）

场景 11

本节主旨 多模态输入带来的 KV Cache 规模

完整内容

江鋆晨：**KV Cache** 该怎么存？它有没有一个更加 scalable 的解决方案？这是个很有意思的问题。大家现在在说长文本，谈的都是文字，包括 code 其实也是文字。

Multimodal 可以处理图像、视频，甚至 3D video。**很多人没有说到的一件事是，图像和视频作为输入时，对于 model 来看其实还是 tokens，但是是 much longer sequence of tokens。**比如几十秒钟的视频，它生成的 token 数可能像一本大部头的书；这几十秒钟视频里面的文字 token 数就已经很多了。视频和 image 变成 token 之后，数量非常庞大。

有了多媒体、multimodal input 之后，包括 video、images，它里面本身自带很多冗余的语义信息，所以这种 input 有很大的压缩空间。这可能是个 research idea：这个 video 不需要整个作为 input，可以先把它转成文字，再作为 input，当然会丢掉一些细节。但是如果知道模型要分析哪些东西，就可以先把它转成文字再给 model，肯定有很多优化空间。

说到前缀复用，或者 context 复用，在 video 和 images 里面，这个复用的定义可能非常不一样。比如同一个 video，480p 和 1080p，同一个 video 的两种不同像素率、清晰度，算不算不同的 input？它可以是一个 input，但转成 token 之后可能长得完全不一样，因为长度都不一样。**所以这里面有很多优化空间。**

所以你觉得就是**存储**硬件本身——

画面说明

画面是江鋆晨的单人访谈近景，底部字幕写着“它里面本身是有自带很多冗余**语义信息**的”；右下角有节目标识，底部导航栏显示“多模态和模型格局等”。

12. 存储供给与未来需求（25:16–25:43）

场景 12

本节主旨 存储供给与未来需求

完整内容

曹卿云：**存储**供给端的这种瓶颈，不会影响未来吗？

江鋆晨：**还会有很多更多的优化技术出现，不会真的影响到实质性的作用。**

江鋆晨：**毕竟需求量会越来越大、越来越大，所以对资源的要求也会越来越高。**你像 NVIDIA、AMD、Intel，年年都会有新的硬件出来，对吧？

我理解就是说，你们的商业本质是建立在算太贵、而存比较便宜的基础上。

画面说明

画面是江鋆晨单人近景，左下角问题条写着“**存储**供应瓶颈会限制未来吗”，底部字幕写着“所以对资源的要求也会越来越高”；右下角有节目标识。

13. 硬件与软件的经济账（25:43–27:10）

场景 13

本节主旨 硬件与软件的经济账

完整内容

曹卿云：所以用存来代算这样的逻辑基础上，存储价格一直在涨，算力当然 GPU 价格也还在涨，我们怎么去 make sense、去算这个经济账呢?

江鋆晨：我们公司并不是做硬件采购生意的。我们是做硬件上面的软件层。但是我们的很多服务对象需要采购这些设备，需要采购 GPU、**存储**设备、CPU，还需要采购这一体机。

我觉得他们现在很多的焦虑是：这个存储价格真的贵。但是大家其实也都知道，存储价格贵是因为供需不太平衡。只要产能上去了，这些 storage 都可以卖得出去，然后价格就会下来。**当然不会下得那么快，因为现在很多存储商的订单都已经定到两年之后了，所以不会下得那么快，但长期来看它是会下来的。**

其实它的经济账应该这样算：我到底应该花多少钱在硬件上，花多少钱在软件上？**因为像我说的，很多优化发生在软件层面，你需要好的工程师帮你做这些优化，或者买好的服务来做这些优化。**

与其花更多钱去买更多 GPU memory，或者买更多 storage 和更贵的 GPU，不如想想是不是可以用软件的方法，让你用现在的 GPU 和现在的 memory 做到类似的事情。其实这里有很大的提高空间。

曹卿云：最终存储的供需平衡还是会有的？**存储还是一个周期，那这次的周期是不是有点不一样？**

画面说明

画面是江鋆晨的单人访谈近景，底部字幕写着“都已经订到两年之后了”；右下角有《Silicon Valley Vector / 硅谷坐标》标识，背景为书架和室内植物。

14. AI 的新使用模式与冷热存储（27:11–29:00）

场景 14

本节主旨 AI 的新使用模式与冷热存储

完整内容

江鋆晨：你像当时做大数据的时候，云厂商在大数据出来之前，它的 balance sheet 可能想的是大家过来租一些机器。但后来有大数据之后，**存储**需要的、算力需要的这些 usage pattern 都完全不一样了。

所以你确实可以说价格会更平衡一点、供需更平衡一点，但是 AI 确实有一个新的 usage pattern，和 cloud 当时不太一样，和大数据也不太一样。这一点我们现在还真看不出来未来会变成什么样。有可能**存储**会一直是个头痛的大问题，也有可能模型会变得越来越多，然后一个 service 需要同时跑 30 个不同的模型，比如 RL training 会把一个模型变成 30 个版本。那样的话 GPU 的用量反而会更高。

所以这些都是 we have to see，不是现在可以直接预测出来的。

曹卿云：如果 KV Cache 是 AIGC 最核心的东西，那听起来这是个长期、需要很长周期的事情。

江鋆晨：对，对，很难看出来存储是不是以后的最大瓶颈。那可能是因为我觉得，毕竟我不希望让大家得买很多存储设备之后才能用我的服务，对吧？KV Cache 是个非常有价值的数据，并不是说存这些东西很贵，而是把它的 insight 挖掘出来。

通过改 KV Cache 和存储 KV Cache 能得到的效果，空间很大，包括可以帮你提速、降本，甚至可以提高 inference quality。

曹卿云：我理解是，你们主要把大量的 KV Cache 存下来，然后把最容易复用的东西放在 HBM、放在这种最热层的存储；把复用比较少、频率比较低的放在更加冷的存储层。这些数据存在越来越冷的层里面。

画面说明

画面是曹卿云的单人近景，底部字幕写着“如果 KV Cache 是 AI 记忆的最核心的东西”；右下角的节目标识在转场中出现部分变形，底部导航栏可见“拆解**存储**需求”等栏目。

15. 冷热分层的计算与存储权衡（29:00–30:41）

场景 15

本节主旨 冷热分层的计算与存储权衡

完整内容

江鋆晨：每存到更冷一层，牺牲的是传输速度。说到底它就是个 trade-off，是计算机系统非常经典的 trade-off，就是计算和**存储**的 trade-off。

传统上，计算省得越多，**存储**就需要越多，然后速度就会降低。在大语言模型里面，很有意思的一件事情是，如果你存的地方够快，比如在 CPU 里面存，甚至在 local SSD 或者 GDS、GPU Direct Storage 里面存这些 **KV Cache**，它可以既让你省掉很多 GPU 计算、减少 GPU cost，同时把这些东西取到 GPU 里面去复用的速度，反而比重新计算要快。

所以它既帮你降本，同时又帮你提速，这是在系统上非常好的事情。当然，如果把 KV Cache 存到更冷、更远端的存储设备上，loading KV Cache 的速度会很慢，所以在某个点上，它会出现你减少 cost、但是 latency 有一定增加的情况。

我们现在看到的很多情况是，因为 KV Cache 本身 size 也会越来越小，它有 compression，所以一般只要网速不要太慢，存储 KV Cache 都是有好处的，并且既可以减少 cost，也可以提高速度。

你如果把这个东西存中国，然后 GPU 跑到美国，那肯定会把整个系统拖得很慢。**但是降本的 benefit 还是在的，因为毕竟让 GPU 少做了很多计算。**

曹卿云：如果未来 KV Cache 要存得更多，你觉得存哪一层可能受益最多，增量最大?

江鋆晨：这是个好问题，这完全取决于每一层。

画面说明

画面是江鋆晨的单人访谈近景，底部字幕写着“存到更冷的更远端的**存储**设备上的话”；右下角有节目标识，底部导航栏可见“**推理**瓶颈”“拆解**存储**需求”等栏目。

16. 热数据、冷数据与 Domain Knowledge（30:42–32:30）

场景 16

本节主旨 热数据、冷数据与 Domain Knowledge

完整内容

江鎏晨：这个价格有多高？你像现在 SSD，价格甚至有可能会接近 RAM。它虽然速度没有高很多，但它可能稀缺，对吧？所以它的价格被推得很高。这样的话，存在 SSD 里面就可能不是那么经济。**但是如果模型特别大、KV Cache 特别大，那必须得用 SSD，SSD 肯定还是有很大的好处。**

我觉得对我们来说，我们的商业模式和我们的技术，跟**存储**技术可以看成两个层面，它并没有耦合的成分。只要你有更多 storage，我们都可以帮你把 **KV Cache** 存下来，而且是以最有效率的方式存下来。

其实你刚刚提到一件事情：**如何保证热的 KV Cache 在快的地方，冷的 KV Cache 在慢的地方？**这件事情是一个非常有意思的系统问题。这个问题在学术界很难解决，在一个产品里面也很难解决。

你必须得把 KV Cache 存在哪里的决定权，给真正的 operator、真正使用这个系统的人，让他们去写：如何判断一个 KV Cache 到底是热的还是冷的？如何判断一个 KV Cache 在下一秒钟会被用，还是下十分钟会被用？如何判断一个 KV Cache 是今天用、明天也用、每天 8 点钟都用？

你把它拆开，就有很多这种优化是产品本身很难去做的。你必须得有一个 interface、一个界面，让正在用你这个系统的人去表达他的 domain knowledge、他知道的这些 knowledge。**这也是我们公司的 vision 之一：我们并不希望做一套 transparent、透明的 KV Cache 系统给这些公司用，而是让这些公司里真正接触这个系统的人，把他们的知识也用在把产品用得更好的过程中。**

曹卿云：我要跟别人说一个比喻，什么样的比喻可以让大家更好地理解 KV Cache 这个——

画面说明

画面是江鋆晨的单人访谈近景，底部字幕写着“给真的用这个系统的人”；右下角有节目标识，画面中可见江鋆晨身穿带 TensorMesh 标识的黑色上衣。

17. 从 CDN 到 KV Cache 分发网络（32:30–35:59）

场景 17

本节主旨 从 CDN 到 KV Cache 分发网络

完整内容

江鋆晨：包括**存储**、包括分发，这件事其实是在 **2000 年**左右非常火的一个概念。当时 CDN 最早和最好的提供商之一叫 Akamai，Akamai 就是当年的 OpenAI。

互联网刚开始的时候，CDN 是一个非常重要的组件。没有它，每个人看网页都得花 10 秒钟甚至更长，才能看到非常基本的网页。有了 Akamai，直接半秒钟就给你了。Akamai 干的事情就是把数据放在用户容易接触到、delay 非常低的地方。

所以我们一开始说 KV Cache 的时候，我就说，是不是应该想一个 CDN? CDN 就是内容分发网络，content delivery network，需要有一个 CDN for **KV Cache**。这样模型需要 KV Cache 的时候，就可以把 KV Cache 从非常近的地方挪过来。

后来发现，现在模型还都是跑在同一个 data center 里面。那么先做一个 data center 内部的 mini CDN，就跟现在做 Kafka storage 差不多。现在大家并没有把 Kafka 分布在全世界不同地方，AI 模型多数也是跑在同一个 data center 里面，那我就在一个 data center 里面做一个类似 mini CDN 的东西。**这就是我们现在做的 KV Cache storage。**

未来趋势不都是 data center 之间也要互相连接吗？其实越往大扩张，这对你们越利好。

江鋆晨：对，我们可能三年前看了六年后的东西。所以有可能过两三年之后，inference service 变得更分布，包括 Edge 出来之后更分布，Edge AI 出来之后更分布式。**更分布式之后，就会需要一套 Internet-scale distribution system，那可能就会回到我们。**

曹卿云：两年前说了，像一个 knowledge delivery network。那为什么现在任务量都只是在一个 data center？**有什么场景可能需要跨数据中心？**

江鋆晨：**有，但是现在这种场景并不多。**现在一般大家还是觉得，在数据中心里面部署 GPU 对 cost 来说最省。同时大家对延迟的要求还没有那么高。毕竟一开始大家做 AI，做 chatbot、chat，只要生成的速度够人看就可以了，对吧？

但是现在是 agent，你生成得越快，agent 跑得越快。真的是生成的东西给它自己读，你生成得当然越快，它读得越快。所以以后大家对 delay 的要求可能会越来越高。对 delay 要求高了之后，就不会把所有东西全跑在同一个地方，大家可能会希望把 AI **推理**系统、这些硬件放到离终端用户越近的地方。离终端用户越近，delay 越低，最后生成的结果就可以让 user 尽快看到，user 可以更好地跟它做 interaction。

等到那个时候，模型离用户越近，模型跟模型之间的距离，或者 engine 跟 engine 之间的距离、GPU 跟 GPU 之间的距离，就会越大。现在是所有人在网络周边，模型和它的硬件全在中间，所有人都跟中间这个东西交流。**但是以后模型可能会离用户越来越近，那模型跟模型之间的距离就会越来越远。**到那个时候，就会有一种新的 CDN 出来。

我们可能当时想得太乐观了，但是三年里面这件事没有发生，不代表六年不会发生，十年不会发生。

曹卿云：**有一个很有意思的事情，你们公司的顾问是 Ion Stoica，他是 Spark 和 Databricks 背后的人。**你为什么选他做你的顾问？你们私下怎么讨论的？

画面说明

画面是曹卿云的访谈近景，左下方姓名条明确写着“曹卿云 / 《硅谷坐标》主持人”，底部字幕写着“都是在一个 data center 的”；右下角有节目标识。

18. 从 AI 大数据谈顾问选择（36:00–37:39）

场景 18

本节主旨 从 AI 大数据谈顾问选择

完整内容

江鎏晨：我们有两个 advisor。**一个叫张辉，张辉以前是 CMU 的教授，现在是 Calybe 的 CEO。**另一个是 Ion Stoica，也是伯克利教授，他是 Databricks founder，也是 Anyscale founder，还有很多公司。

他们俩其实跟我都是同一个组的。**以前 Ion 是张老师很早时候的学生，我是张老师最后一个学生，大家以前合作很多。**

为什么找 Ion？是因为我们一直觉得，我们做的事情并不是纯粹的 GPU 计算问题。我们很早就一直说，**KV Cache** 是 AI 的大数据。**所以我们一直有一个 knowledge：这个东西跟 Databricks 有什么关系？**非常高层次上，可能是有相似性的。

他关注 AI inference。你知道，他的组是做 SGLang 和 vLLM 项目的组，所以他们对这方面非常懂。我跟 Ion 谈 **KV Cache** 的时候，他其实一开始也没听懂我在说什么。但我说得多了以后，他也意识到：这事真的有一些是个数据问题，跟 vLLM 做的那些计算问题不是一回事，这是个数据问题，是个数据公司。

关于我们做的事情和 big data、大数据的关系，他其实也提了很多建议，他也看得到一个相似性。当然，Databricks 做的东西跟我们做的东西其实还是两个世界。**Databricks 做的是企业级用户数据，human-readable data，你可以这样说。**

江鋆晨：我们做的是只有 model 才能看到的 data。**如果有一个世界全是 AI model，那可能我们干的是 Databricks 的事。**哈哈哈。

曹卿云：你怎么看最近的 token maxing? **而且从 token maxing 到 token 的 utilization efficiency，你觉得里面最大的机遇和挑战是什么?**

画面说明

画面是主持人与嘉宾的双人访谈全景，底部字幕写着“我跟 Ion 谈这个 **KV Cache** 的时候”；右下角有节目标识，场景中可见两人、书架、窗户和中间的小桌。

19. Token Maxing 与非前缀复用（37:39–40:03）

场景 19

本节主旨 Token Maxing 与非前缀复用

完整内容

江鋆晨：**我对 token maxing 的粗浅理解是，大家对于 AI model 的能力有很强的自信，所以把所有东西都放到 AI model 上。**越来越复杂的任务放到 AI model 上去做，token 的消耗量就会越来越高、越来越高。

并且有时候 AI 做了一个错误的决定，你还需要让 AI 去 fix，对吧？投入越来越多。已经有很多公司意识到这个事情：这个 cost 并不能 justify 它的 benefit。但这不代表技术不行，技术还是有用的，只不过需要有更好的方法来利用这个技术。

这也是为什么现在很多 agent 公司非常非常火。它们在用什么方式解决问题呢？它们告诉你哪些 tokens 或者哪些 context 是真的有用，**如何发掘真正有用的 context，**如何跟工程师有更好的 interface，让工程师更好地 express their intention，然后可以减少 token 的用量。

你像我说的，jellystone，你把一个东西削得越高，大家需求上只会越短、越高。**最后对于我们来说，结果就是 token consumption 的量只会越来越高、越来越高。**

最有意思的事情是，现在大家做 token maxing 的时候，都是以野蛮生长的方式去吸引这些 agent。所谓野蛮生长，就是模型看过什么、模型生成什么，就直接放到 context 里面，下次继续让它读。**相当于每次问模型的时候都说：这是你之前看到的所有东西，和你之前说过的所有话，读一遍，回答新的问题。**

这真的是非常非常 naive 的方法。先不管 model quality 怎么样，这非常消耗 tokens。所以有很多公司在做 compression，把以前的东西 compact 一下或者什么的。**但是毕竟很多时候东西没法 compress，它只能让它复用；而复用不需要复用所有东西，这就是我们做技术有用的地方之一。**

野蛮生长的时候，大家复用的都是前缀。**但是聪明一点的时候，会有很多复用是非前缀的东西。**就是回到刚刚说的 CacheBlend 这个技术，有很多公司现在用 CacheBlend，对这个技术特别感兴趣，就是因为——

曹卿云：**可以让他们做更有效率的 token maxing，不需要像模型这样一直读以前所有东西、再看一遍。**我们想谈谈现在大模型公司的格局，你的看法是什么？你未来希望这些大模型公司是一家独大，还是百花齐放？哪个对你们的发展更好？

画面说明

画面是主持人与嘉宾的双人访谈全景，底部字幕写着“最有意思的事情是”；右下角有节目标识，底部导航栏可见“**KVCache 三层理解**”和“客户画像”等栏目。

20. 大模型公司的两种商业结局（40:03–41:32）

场景 20

本节主旨 大模型公司的两种商业结局

完整内容

江鋆晨：**最后会有两种商业结局吧。**

第一种商业结局，终局像 search 一样，搜索是 Google 一家独大，对吧？其他的份额都很小，小到多数你都听不到，大公司也都用 Google。

第二种结果像视频、像视频软件一样。**虽然 YouTube 是最好的、最大的网站，当然其次还有 Netflix，但是你有很多叫得上名、也非常重要的视频网站。**比如 HBO、MSNBC、BBC，或者 Hulu，有很多叫得上名、很多人都离不开的视频网站，对吧？这都是非常成功的商业模式。

我觉得大模型更像在线视频，会有少数公司非常非常大、consolidate，但是会有很多 service 需要自己独立的生存空间，并且也非常重要，有自己的生态。

有很多情况不能用第三方闭源模型这种服务。**比如主权 AI、客户信息或者商业机密保护得非常好的企业级用户，他们永远需要一套自己可以维护，或者自己可以控制的软件系统。**我猜它最后更像 online video 的结果。

曹卿云：中国最近的这些模型表现也挺好。不管是 **25 年**的 DeepSeek，还是 **26 年**最近的智谱 GLM，表现都挺惊艳的，你是怎么看的?

画面说明

画面是江鋆晨的单人访谈近景，底部字幕写着“会有少数的公司会非常大”；右下角有《Silicon Valley Vector / 硅谷坐标》标识，画面中可见 TensorMesh 标识的黑色上衣。

21. 中国开源模型与 Transformer 架构（41:33–42:18）

场景 21

本节主旨 中国开源模型与 Transformer 架构

完整内容

江鋆晨：首先，它们表现的 quality 越来越好，背后有很多原因，这就不深入讲了。**但是效果、结果确实是，美国的大公司都在用中国的开源模型，对吧？**因为这些模型确实是开源模型里面效果最好的。

你像 OpenAI 也开源过模型，Google 也开源过模型，但这些模型最后或多或少很难和中国的这些开源模型竞争。所以我觉得——

曹卿云：除非有很大的改变，不然的话，我觉得未来几年里面，开源模型可能还得看国内的开源模型。现在大多数模型都是建立在可以产生 **KV Cache** 的 Transformer 架构上，如果以后换成新的、比如 Mamba 的架构，它产生的状态不一定是 **KV Cache**，这对你——

画面说明

画面是曹卿云的单人访谈近景，底部字幕写着“但这些模型最后都或多或少很难和”；右下角有节目标识，底部导航栏可见“多模态和模型格局等”。

22. Mamba、Transformer 与 Hybrid Model（42:18–43:40）

场景 22

本节主旨 Mamba、Transformer 与 Hybrid Model

完整内容

曹卿云：们的影响会怎么样?

江鋆晨：首先，我们一直说的 **KV Cache**，其实是一个暂时的名字。我们所谓 **KV Cache** 这个概念，是 model-native data、AI-native data。

这种 data 是什么呢？是模型在**推理**过程中，或者说在运行过程中生成的中间状态，生成它自己内部的理解。这种数据其实在 Mamba 里面也有，在 diffusion model 里面也有，在以前的 convolutional neural network 里面也有，所有 model 都有，只不过现在这个阶段 Transformer 是最 popular 的 architecture。

然后，KV Cache 是 Transformer 的 AI-native data，所以我一直说 KV Cache。Mamba 确实跟 Transformer 有很大的不同，它所谓的 intermediate、所谓的 AI-native data，是一个 linear state，是一个跟文本长度不直接相关的 size。

首先 Mamba 并不是现在用得很多，只不过很多模型在用 Mamba 的架构去替换中间的某些层。像千问 3.5，它也是多数层是 Mamba，但仍有不少层，大概四分之一的 layers，还是用 full attention、还是 Transformer。

所以现在业界的共识是，用一些 Mamba、用一些 Transformer，把它们结合起来，所以叫 hybrid model。

曹卿云：最近 xAI 收购了 Cursor，你是怎么看的？xAI 收购 Cursor。

画面说明

画面是主持人与嘉宾的双人访谈全景，底部字幕写着“那 Mamba 确实跟 transformer 有很大的不同”；右下角有节目标识，底部导航栏可见“多模态和模型格局等”。

23. 推理服务的整合趋势（43:40–44:36）

场景 23

本节主旨 推理服务的整合趋势

完整内容

江鋆晨：现在其实是个趋势：这些**推理**厂商，或者**推理**相关的，包括 agent、底下的模型，到 service provider，其实大家都在 consolidate。

这个 consolidation 肯定会发生。就和当时做 cloud、做 container service 或者做 big data 一样，一开始都是百花齐放，但很快大家都会 consolidate。

Cursor 对于 xAI 来说，本身是有商业价值的。**但是这只不过是现在很多收购和并购的例子之一。**我说实话，我对这个具体的例子并不是特别了解。现在每个月都可以听到一个比较大的并购。

曹卿云：和收购，以后可能每个礼拜都会听到。今天最后一个问题，我们团队觉得 **KV Cache** 是 AI 时代的汽油。因为在提炼石油的时候，有一个副产品是汽油，这个汽油长期来说被人当作一个废弃的燃料，直到内燃机出现，它才被重视——

画面说明

画面是主持人与嘉宾的双人访谈全景，没有额外图表或产品画面；两人坐在室内书架和窗户前，右下角有《Silicon Valley Vector / 硅谷坐标》标识。

24. KV Cache 是 AI 时代的汽油（44:36–45:23）

场景 24

本节主旨 KV Cache 是 AI 时代的汽油

完整内容

江鋆晨：并且是驱动了整个现代工业。不知道你觉得这个 **KV Cache** 未来会走一样的路吗?

这个比喻非常好。确实绝大多数人在看 **KV Cache** 的时候，大家觉得 KV Cache 就是模型在运行当中生成的一个 by-product，对吧？在 GPU 内部存一会儿之后就丢了。

当然，对于做 ML 的人来说，他们知道 KV Cache 是非常有意思的信息，但是他们接触不到 KV Cache。**所以现在其实是：懂 KV Cache 的人接触不到它，接触得到 KV Cache 的人不懂 KV Cache。**但是哪天真的有一个公司把这个价值挖出来了，它就从废料变宝了。

曹卿云：感觉跟汽油一样。**你觉得未来这个内燃机的时刻是什么时刻?**

江鋆晨：**取决于我们公司的发展。**

曹卿云：**太好了，非常期待你们的 KV Cache moment，你们的内燃机时刻。**感谢今天的时间。

画面说明

画面是江鋆晨的单人访谈近景，左下方问题条明确写着“KVCache 是 AI 时代的汽油”，底部字幕写着“所以其实现在是懂 KVCache 人接触不到”；右下角有节目标识。