Under The Hood: How OpenAI's Sora Model Works

Matthias Plappert

15 Mar 2024

OpenAI's Sora model has amazed the world by its ability to generate extremely realistic videos of a wide variety of scenes. Below is a video released by OpenAI that demonstrates the capabilities of the model.

Video from "Introducing Sora - OpenAI's text-to-video Model"

In this blog post, we dive into some of the technical details behind Sora. We also talk about our current thinking around the implications of these video models. Finally, we discuss our thoughts around the compute used for training models like Sora and present projections for how that training compute compares to inference, which has meaningful indications for estimated future GPU demand.

Key Findings

The key findings from this report are summarized below:

- Sora is a diffusion model that builds on top of Diffusion Transformers (DiT), Latent Diffusion and appears to scale up both the model and the training dataset significantly.
- Sora demonstrates that scaling up video models is worthwhile and that further scaling, similar to Large Language Models (LLMs), will be the main driver for rapidly improving models.
- Companies like Runway, Genmo and Pika are working on building intuitive interfaces and workflows around video generation models like Sora. This will determine how widely useful and usable they become.
- Sora requires a huge amount of compute power to train, estimated at 4,200-10,500 Nvidia H100 GPUs for 1 month.
- For inference, we estimate that Sora can at most generate about 5 minutes of video per hour per Nvidia H100 GPU. Compared to LLMs, inference for diffusion-based models like Sora is multiple orders of magntitude more expensive.

> TikTok (50% of all video minutes) and YouTube (15% of all video minutes) and taking hardware utilization and usage patterns into account, we estimate a peak demand of ~720k Nvidia H100 GPUs for inference.

In summary, Sora demonstrates major progress in quality and capabilities for video generation, but also has potential to greatly increase demand for GPU inference compute.

Background

Sora is a diffusion model. Diffusion models are a popular choice for image generation and well-known models like OpenAI's DALL-E or Stability AI's Stable Diffusion. More recently, companies like Runway, Genmo, and Pika have explored video generation, likely utilizing diffusion models as well.

Broadly speaking, diffusion models are a type of generative machine learning model that learns to create data resembling the data they were trained on, such as images or video, by gradually learning to reverse a process that adds random noise to data. Initially, these models start with a pattern of pure noise and step-by-step remove this noise, refining the pattern until it transforms into coherent and detailed output.

Illustration of the diffusion process: Noise is gradually removed step-by-step until the detailed video is visible. Image taken from the Sora technical report.

This is notably different from how Large Language Models (LLMs) work conceptually: LLMs iteratively produce tokens[1] one after another (this is called autoregressive sampling). Once a token has been produced, it will not be changed. You have likely seen this effect in action when using tools like Perplexity or ChatGPT: The answer gradually appears word by word, as if someone is typing.

paper, in which the authors[2] propose a Transformer-based architecture called DiT (short for *Di*ffusion *T*ransformers) for image generation. It appears that Sora extends this work to video generation. Combining both the Sora technical report and the DiT paper, we can thus get a fairly accurate picture of how the Sora model works.

There are three important parts to Sora: 1) It does not operate in pixel space but instead performs diffusion in latent space (aka latent diffusion), 2) it uses the Transformer architecture, and 3) it appears to use a very large dataset.

Latent Diffusion

To understand the first point, latent diffusion, consider generating an image. You could generate each pixel using diffusion. However, this is highly inefficient (a 512×512 image has 262,144 pixels, for example). Instead, you can first map from pixels to a latent representation with some compression factor, perform diffusion in this more compact latent space, and finally decode back from latent into pixel space. This mapping significantly improves the computational complexity: instead of having to run the diffusion process over 512×512 = 262,144 pixels, you only have to generate 64×64 = 4,096 latents, for example. This idea was the key breakthrough in the "High-Resolution Image Synthesis with Latent Diffusion Models" research paper, which is the foundation of Stable Diffusion.

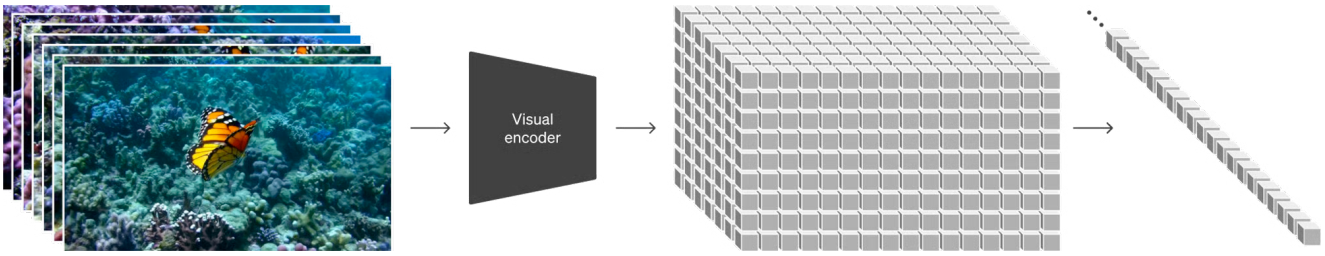


Illustration of the mapping from pixels (left) to a latent representation (the grid of boxes on the right). Image taken from the Sora technical report.

Both DiT and Sora utilize this approach. For Sora, an additional consideration is that videos have a temporal dimension: A video is a temporal sequence of images, also called frames. From the Sora technical report, it appears that the encoding step that maps from pixel to latent space happens both spatially (meaning compressing the width and height of each frame) and temporally (meaning compressing across time).

Transformers

Illustration how model quality improves as a function of training compute: base compute, 4x compute, and 32x compute (from left to right). Videos taken from the Sora technical report.

This scaling behavior, which can be quantified by so-called scaling laws, is an important property and it has been studied before in the context of Large Language Models (LLMs) and for autoregressive models on other modalities. The ability to apply scale to obtain better models was one of the key drivers behind the rapid progress on LLMs. Since the same property exists for image and video generation, we should expect the same scaling recipe to work here, too.

Dataset

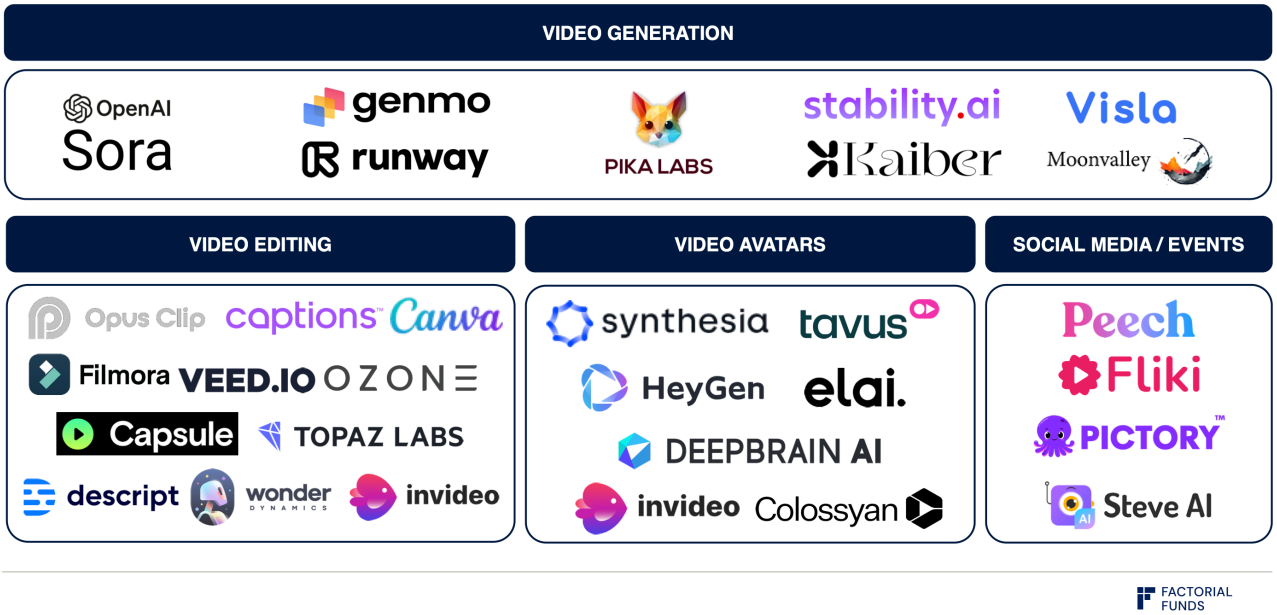
The final key ingredient that is required to train a model like Sora is labeled data and we think this is where most of the secret sauce is. To train a text-to-video model like Sora, you need pairs of videos and textual descriptions thereof. OpenAI does not talk much about their dataset but they hint that it is very large: "*We take inspiration from large language models which acquire generalist capabilities by training on internet-scale data.*" (source). OpenAI has further published a method for annotating images with detailed text labels, which was used to collect the DALLE-3 dataset. The general idea is to train a captioner model on a labeled subset of your dataset and to use that captioner model to automatically label the rest. It appears that the same technique was applied for Sora's dataset.

Implications

We believe that Sora has a few important implications. We'll briefly discuss those now.

Video models are starting to be actually useful

The quality of the videos Sora can generate is clearly a breakthrough both in terms of the level of detail but also in terms of temporal consistency (for example, the model



Market map of companies in the video generation space.

There are remaining challenges though: It's currently unclear how steerable Sora models are. Editing a generated video is difficult and time consuming since the model outputs pixels. And building intuitive UIs and workflows around these models is also necessary to make them useful. Companies like Runway, Genmo and Pika and many more (see above market map) are already working on these problems.

Scaling works for video models so we expect rapid progress

A key insight of the DiT paper was that model quality directly improves with additional compute, as discussed above. This is similar to the scaling laws that have been observed for LLMs. We should therefore expect rapid further progress on the quality of video generation models as these models are trained with more and more compute. Sora is a clear demonstration that this recipe indeed works and we expect OpenAI and others to double down on this.

Synthetic Data Generation and Data Augmentation

In domains like robotics and self-driving cars, data is inherently scarce: There is no internet full of robots doing tasks or cars driving. Typically, these problems have therefore been approached by either training in simulation or by collecting data at scale in the real world (or a combination of both). However, both approaches struggle since simulated data is often unrealistic. Collecting real-world data at scale is

Illustration of augmenting a video by modifying some of its properties, in this case rendering the original video (left) in a lush jungle setting (right). Image taken from the Sora technical report.

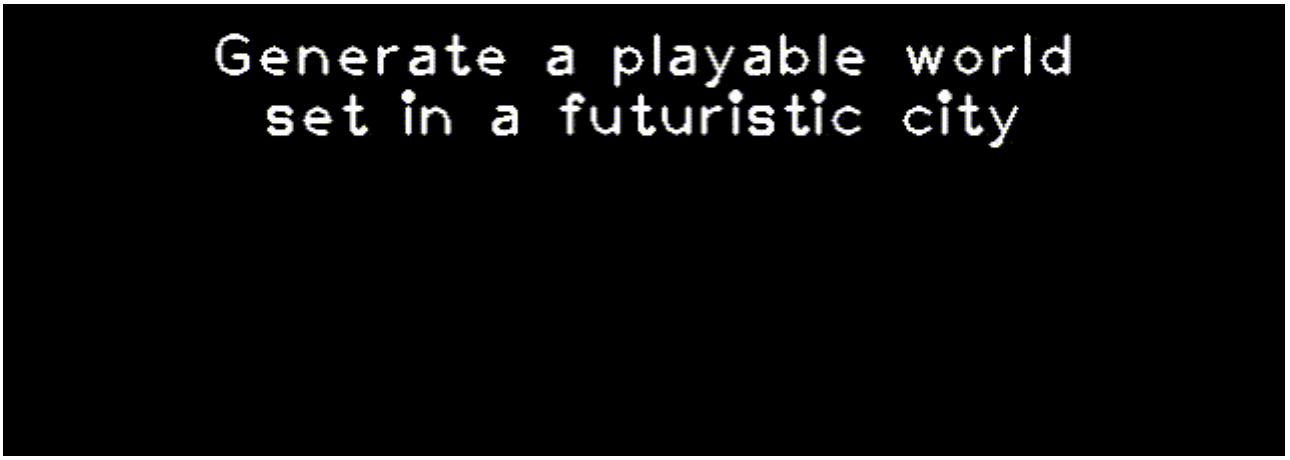
We believe that models like Sora can be very useful here. We think it's possible that Sora-like models could be used to generate fully synthetic data directly. Sora can also be used for data augmentation where existing video is transformed into different appearances. This second point is illustrated above where Sora converts a video of a red car driving on a forest road into a lush jungle scenery. You could imagine using the same technique to re-render scenes at day vs. night or to change the weather conditions.

Simulations and World Models

A promising research direction is to learn so-called world models. If sufficiently accurate, these world models allow one to train agents directly within them or they might be used for planning and search.

It appears that models like Sora implicitly learn a basic simulation of how the real world works directly from video data. This "emergent simulation" is currently flawed but it is exciting nonetheless: It suggests that we might be able to train these world models at scale from video. Furthermore, Sora appears to be able to simulate very complex scenes like liquids, reflections of light, fabrics and hair movement. OpenAI even titled their technical report "Video generation models as world simulators", which makes clear that they believe this to be the most important aspect of their model.

Very recently, DeepMind has demonstrated a similar effect with their Genie model: By training on only videos of video games, the model learns to simulate these games (and comes up with new ones). In this case the model even learns to condition on actions without observing them directly. Again, the goal is to enable learning directly in these simulations.



Video from Google DeepMind's "Genie: Generative Interactive Environments" introduction.

Combined, we believe that models like Sora and Genie might turn out to be extremely useful to finally train embodied agents (e.g. in robotics) on real-world tasks at scale. There are limitations though: since these models are trained in pixel space, they model every detail like how the wind moves leaves of grass, even if that is completely irrelevant to the task at hand. While the latent space is compressed, it still has to retain a lot of this information since we need to be able to map back to pixels, so it is unclear if planning can efficiently happen in this latent space.

Compute Estimates

At Factorial Funds, we like to look at how much compute was used both for training and for inference. This is useful since it can inform forecasts of how much compute will be needed in the future. However, estimating these figures is also difficult to do since there are very few details available on the model size and dataset used to train Sora. **The caveat is thus that the estimates in this section are highly uncertain, so they should be taken with a grain of salt.**

Extrapolating training compute from DiT to Sora

Details on Sora are very thin but we can again look at the DiT paper, which is clearly the foundation for Sora, and extrapolate the compute figures presented there. The largest DiT model, DiT-XL, has 675M parameters and was trained with a total compute budget of approximately 10^{21} FLOPS.[3] To make this number easier to understand, this is equivalent to approximately 0.4 Nvidia H100s for 1 month (or a single H100 for 12

up with 180 frames in latent space. So we obtain a compute multiplier of 180x over DiT when naively extrapolating it to videos.

We further believe that Sora is significantly larger than 675M parameters. We estimate that a 20B parameter model is feasible, which gives us another 30x in compute over DiT.

Lastly, we believe that Sora was trained on a much larger dataset than DiT. DiT was trained for 3M training steps at batch size 256, i.e. on a total of 768M images (note though that same data was repeated many times since ImageNet only contains 14M images). Sora appears to have been trained on a mixture of images and videos but beyond that we know almost nothing about the dataset. We therefore make the simple assumption that Sora's dataset is 50% still images and 50% videos and that the dataset is 10x-100x larger than the one used by DiT. However, DiT repeatedly trained on the same data points, which is likely suboptimal if a much larger dataset is available. We therefore believe that a compute multiplier of 4-10x is a more reasonable assumption.

Putting the above together and considering both the low and high estimate for the additional dataset compute, we arrive at the following calculation:[4]

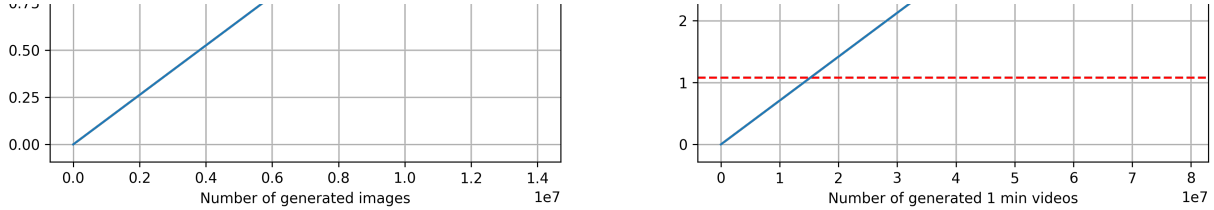
- Low dataset estimate: 10^{21} FLOPS × 30 × 4 × (180 / 2) ≈ 1.1×10^{25} FLOPS
- High dataset estimate: 10^{21} FLOPS × 30 × 10 × (180 / 2) ≈ 2.7×10^{25} FLOPS

This corresponds to 4,211 - 10,528 Nvidia H100s for 1 month.

Inference vs. Training Compute

Another important consideration that we tend to look at is how training compute compares to inference compute. Conceptually, training compute is very large but is also a one-off cost that is incurred once. In contrast, inference compute is much smaller but is incurred for every single generation. Inference compute thus scales with the number of users and becomes increasingly important as a model gets widely used.

It is therefore useful to look at the "break-even point", i.e. the point at which more compute is spent on inference than was spent during training.



Comparison of the training vs. inference compute for DiT (left) and Sora (right). For Sora, our data is based on the above estimate and thus not entirely reliable. We also show two estimates for training compute: one low estimate (assuming a 4x multiplier for dataset size) and a high estimate (assuming a 10x multiplier for dataset size).

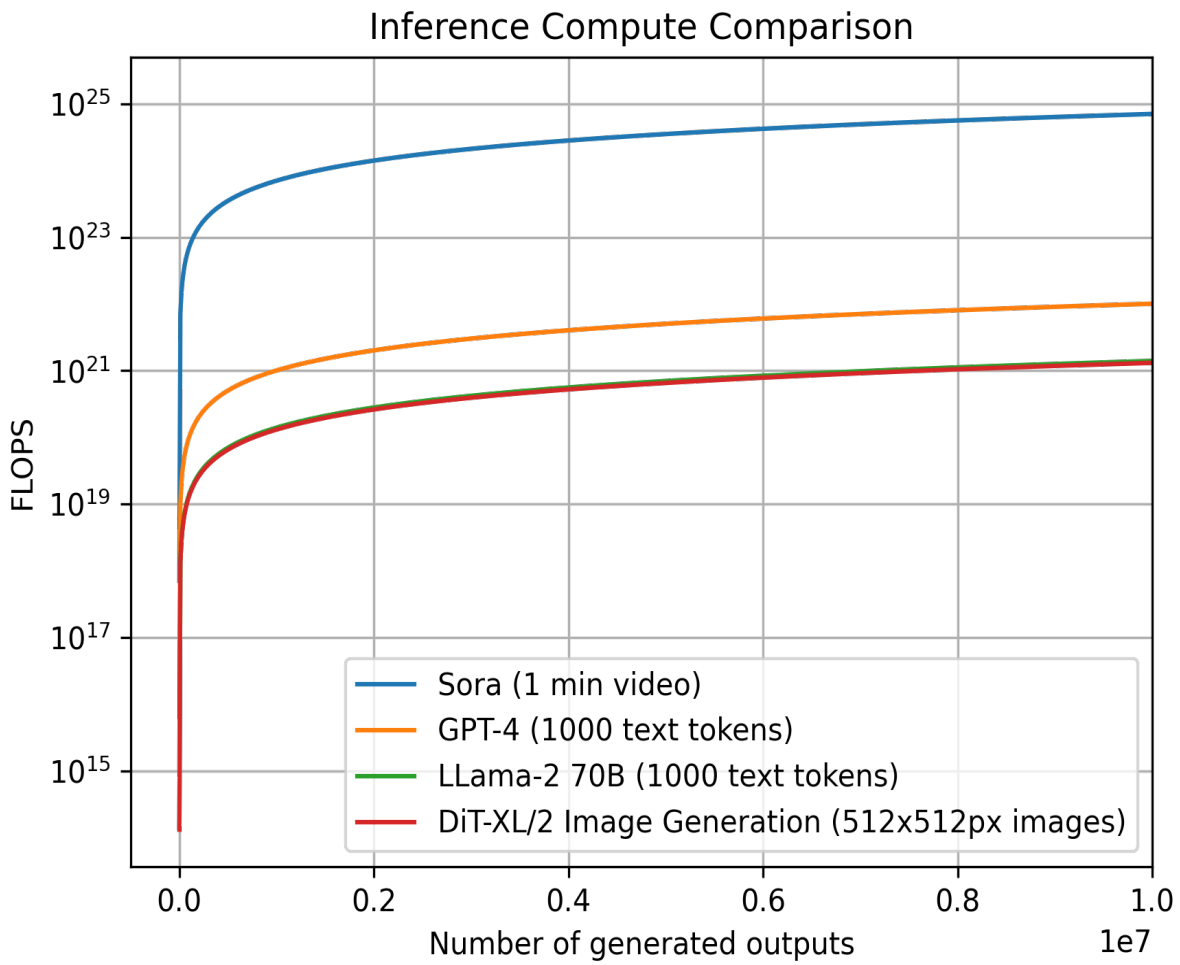
For the above figures, we again use DiT to extrapolate to Sora. For DiT, the largest model (DiT-XL) uses 524×10^9 FLOPS per step and DiT uses 250 diffusion steps to generate a single image, for a total of 131×10^{12} FLOPS. We can see that the break-even point is reached after generating 7.6M images, after which the inference compute dominates. For reference, users upload roughly 95M images *per day* to Instagram (source).

For Sora, we extrapolate the FLOPS to 524×10^9 FLOPS × 30 × 180 ≈ 2.8×10^{15} FLOPS. If we still assume 250 diffusion steps per video, that's a total of 708×10^{15} FLOPS per video. For reference, that's about 5 minutes of video generated per Nvidia H100 per hour.[5] The break-even point is reached after either 15.3M (low) to 38.1M (high) minutes of video generated, after which more inference than training compute is spent. For reference, about 43M minutes of video are uploaded to YouTube *per day* (source).

A few caveats: For inference, FLOPS is not the only aspect that matters for inference. Memory bandwidth is another important factor, for example. Furthermore, there is active research on reducing the number of diffusion steps, which leads to potentially much less compute-intensive and therefore much faster inference. FLOPS utilization rates can also vary between training and inference, in which case they become important to consider.

Inference Compute Across Different Models

We also look at how inference compute per unit of output behaves across different models for different modalities. The idea here is to see how much more compute intensive inference is for different categories of models, which has immediate



Comparison of inference compute by model per unit of output (for Sora, 1 min video, for GPT-4 and LLama 2 1000 tokens of text, and for DiT a single 512×512px image). We can see that our estimate for Sora's inference is orders of magnitude more computationally expensive.

We compare Sora, DiT-XL, LLama 2 70B and GPT-4 and plot them against each other (using a log-scale for FLOPS). For Sora and DiT, we use the inference estimates from above. For Llama 2 and GPT-4, we estimate the number of FLOPS using the rule-of-thumb formula of FLOPS = 2 × number of parameters × number of generated tokens. For GPT-4, we assume the model is a Mixture of Experts (MoE) model with 220B parameters / expert and 2 experts active per forward pass (source). Note that for GPT-4, these figures are not confirmed by OpenAI, so they again need to be taken with a grain of salt.

We can see that inference for diffusion-based models like DiT and Sora is much more

utilization of GPUs, limitations around memory capacity and memory bandwidth, and advanced techniques like speculative decoding.

Inference compute if Sora-like models achieve significant market share

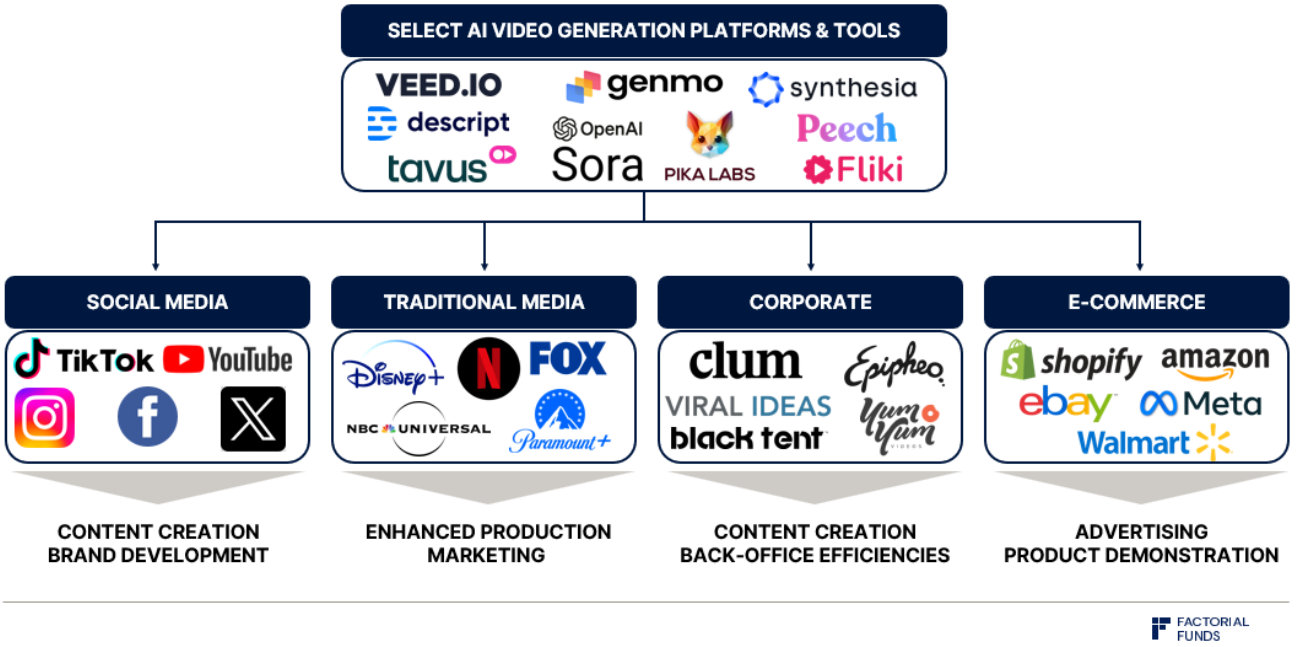
In this section, we extrapolate from Sora's compute requirements to see how many Nvidia H100s would be needed to run Sora-like models at significant scale, meaning that AI-generated videos achieve a significant market penetration on popular video platforms like TikTok and YouTube.

- We assume 5 minutes of videos produced per Nvidia H100 per hour (see above for details), equivalent to 120 minutes of videos per H100 per day
- TikTok: 17M minutes videos per day (34M total videos × avg. length of 30s), assuming 50% penetration by AI (source)
- YouTube: 43M minutes videos per day, assuming 15% penetration by AI (mostly video below 2 min)
- Total videos produced daily by AI: 8.5M + 6.5M = 10.7M minutes
- **Total Nvidia H100 needed to support the creator community on TikTok & YouTube: 10.7M / 120 ≈ 89k**

This figure is likely too low due to various factors that need to be accounted for:

- We assume 100% FLOPS utilization and do not consider memory and communication bottlenecks. In reality a utilization of 50% is more realistic, which adds a factor of 2x.
- Demand is not distributed equally across time but instead is bursty. Peak demand is especially problematic since you need proprotionally more GPUs to still serve all traffic. We think that peak demand adds another factor of 2x for the maximum number of GPUs needed.
- Creators will likely generate multiple candidate videos to select the best one from these candidates. We make the conservative assumption that on average 2 candidates for each uploaded video are generated, which adds another factor of 2x.
- **In total this leaves us with ~720k Nvidia H100 GPUs at peak**

This demonstrates our believe that inference compute will dominate as generative AI models become increasingly popular and relied upon. For diffusion-based models like Sora, even more so.



Illustrative video content creation use cases driving what is expected to be the most immediate demand for models such as OpenAI's Sora

---

We're always looking for feedback to improve our thinking. Please feel free to reach out to matthias.plappert@factorialfunds.com with any comments you might have.

Footnotes

1. A “token” can approximately be understood to be a single word. ↩
2. This paper was co-authored by William Peebles, who has since been hired by OpenAI and is one of the lead authors of the Sora technical report. ↩
3. See “Scalable Diffusion Models with Transformers”, Fig. 9. ↩
4. The formula is: base compute for DiT × model size increase × dataset size increase × compute increase due to 180 frame video data but only for 50% of the dataset. ↩
5. Ignoring memory constraints and only considering FLOPS. ↩
6. For reference, the average Wikipedia article has about 670 words per article. ↩

Share in social media:

LOS ANGELES

1925 Century Park E,
Los Angeles, CA 90067

SAN FRANCISCO

345 California St
San Francisco, CA 94104

EMAIL

contact@factorialfunds.com