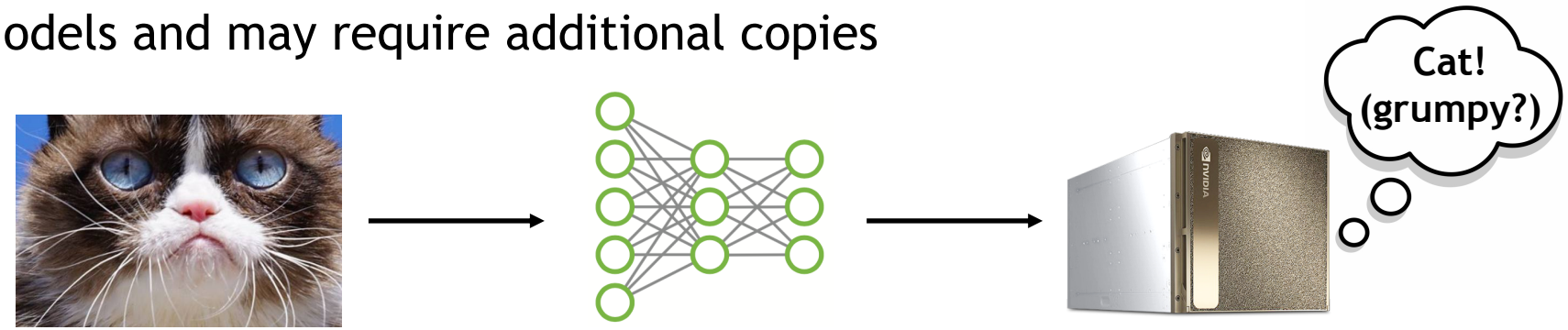

# Vulkan ML

Piers Daniell
July 2019

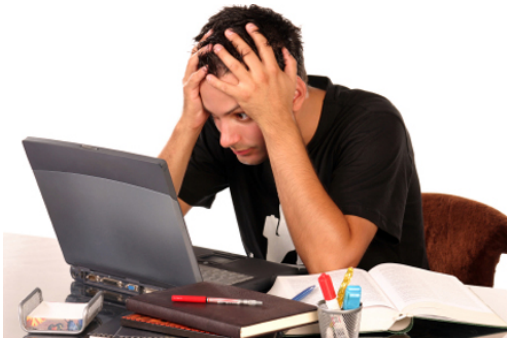
# Why machine learning in Vulkan?

- **Research showcases potential use of machine learning in interactive and high frame rate applications**
  - Character animation (phase function neural network, etc.)
  - Full screen image processing (antialiasing, upscaling, inpainting, DLSS, etc.)
  - Non-Player Character bots (AlphaStar, OpenAI Five, etc.)
  - Image generations (GAN, fire & smoke & clouds, etc.)
- **Current machine learning solutions have relatively high interop overhead**
  - Interop with third party framework (Python TensorFlow, PyTorch, OpenCL, etc.) introduces bubbles where the CPU/GPU are not doing useful work
  - Sharing data with external APIs can be challenging due to difference in memory models and may require additional copies



# How to do machine learning in Vulkan?

- **This is possible already today**
  - Just use compute shaders to implement the various algebra operations.
  - Examples: Tencent/ncnn, Alibaba/MNN, Unity ML, etc.
- **Or use compilers which will generate SPIR-V code for you**
  - Examples: TVM.AI
- **But**
  - Writing high efficiency layered matrix multiplications, with various activation functions requires some **advanced GPU programming skills**, with **different solutions for different hardware**



# The Vulkan ML TSG (Technical Subgroup)

- **A new technical subgroup at Khronos has been formed to improve the solution space for machine learning in Vulkan**
- **Includes representatives from many companies**
- **Goals**
  - Investigate proprietary extensions for inclusion into core Vulkan (**VK_NV_cooperative_matrix**, etc.)
  - Improvements to compute shaders specific to ML needs
  - New cross vendor extensions (meta-commands, etc.)
- **If you are interested, please reach out to us: pboudier@nvidia.com**

# VK_NV_cooperative_matrix

- Exposes NVIDIA's Turing Tensor Cores to Vulkan/SPIR-V
- Accelerates large, low-precision matrix multiplies
- Core compute function for deep learning
- FP16 supported today, UINT8/SINT8 coming soon
- Sample code: https://github.com/jeffbolznv/vk_cooperative_matrix_perf
- Performance on NVIDIA TITAN RTX
  - fp16 matrix math with fp16 accumulation: **100 TFLOPS**