Praktické zkušenosti s LLM na platformě Microsoft

Václav Kyba
Miroslav Janata
SKILL s.r.o.

Úvod

- **Proč na Microsoftu ?**
- **Zkušenosti se zprovozněním a následným provozem LLM**
- **Problémy .... , slepé uličky**
- **Funkční konfigurace**
- **Otázky ?**

Proč na Microsoftu ?

- **IS pro zpracování UI** Zákon č. 412/2005 Sb., o ochraně utajovaných informací a o bezpečnostní způsobilosti ukládá v § 34 povinnost používat pro nakládání s utajovanými informacemi informační systémy certifikované Národním úřadem pro kybernetickou a informační bezpečnost (**NÚKIB**) + Nařízení vlády č. 522/2005 Sb. - seznam utajovaných informací
- **NÚKIB - Podpora OS v certifikovaných IS** (podpora pouze pro Windows, linux nebo ostatní systémy prakticky 0)
- **OS založené na Linuxu či jiném systému = spoustu trpělivosti, mraky dokumentace a, … mnoho, mnoho času.** Alternativa jak v certifikovaném systému provozovat OS na linuxu je provozovat jej jako Appliance – blackbox. Jak ale zajistit zákazníkovi podporu takého systému ?
- **Microsoft = Ušetříte si mnoho problémů sobě a také budoucímu provozovateli systému**

Zkušenosti se zprovozněním LLM

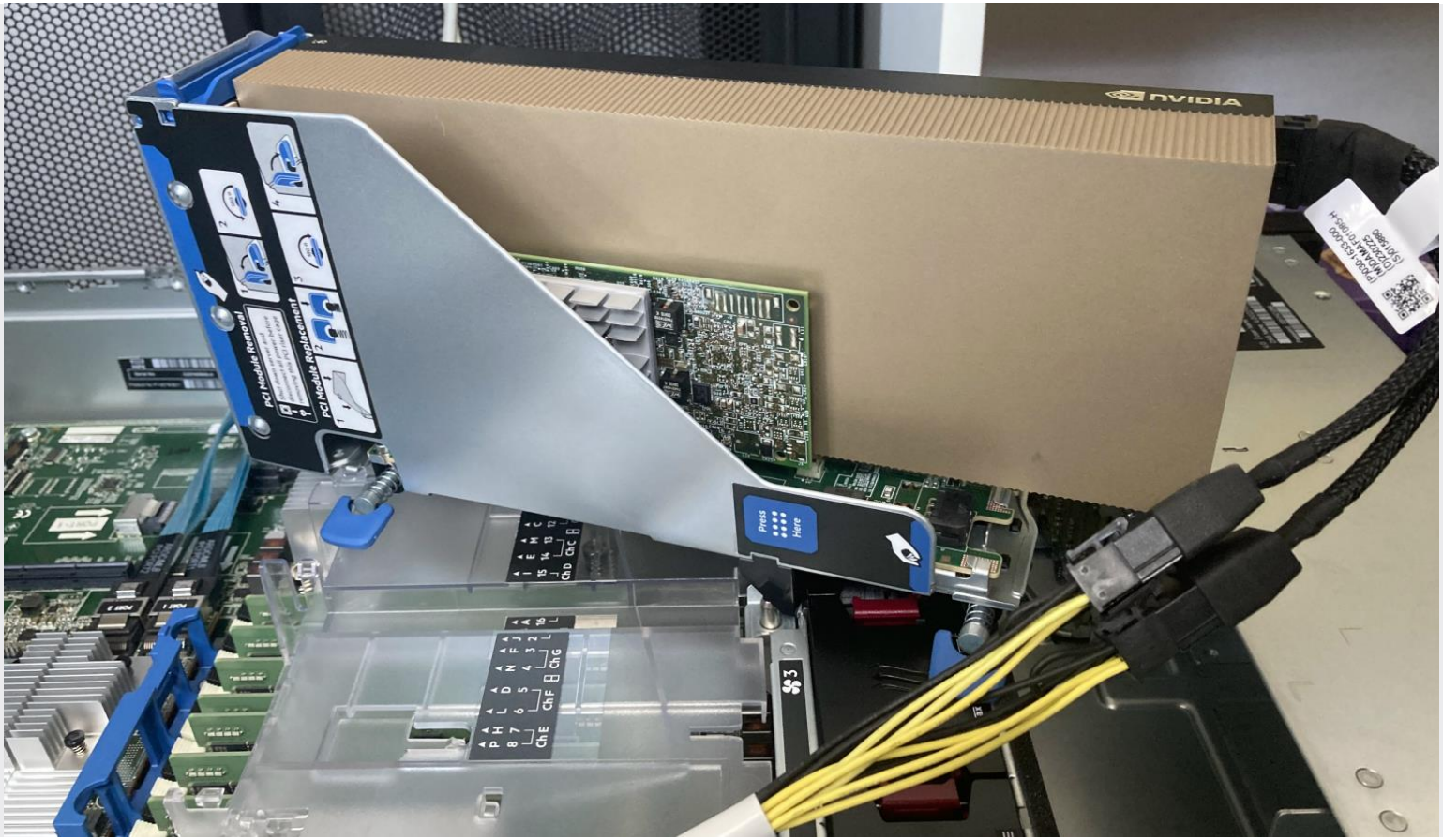
- **HW – HPE DL385 G10** (3 roky staré, AMD Epyc, dnes Gen11)

  P/N P14278-B21, RAM 512GB 3200, 2x CPU AMD EPYC 7282 16-Core Processor / 2800 MHz.

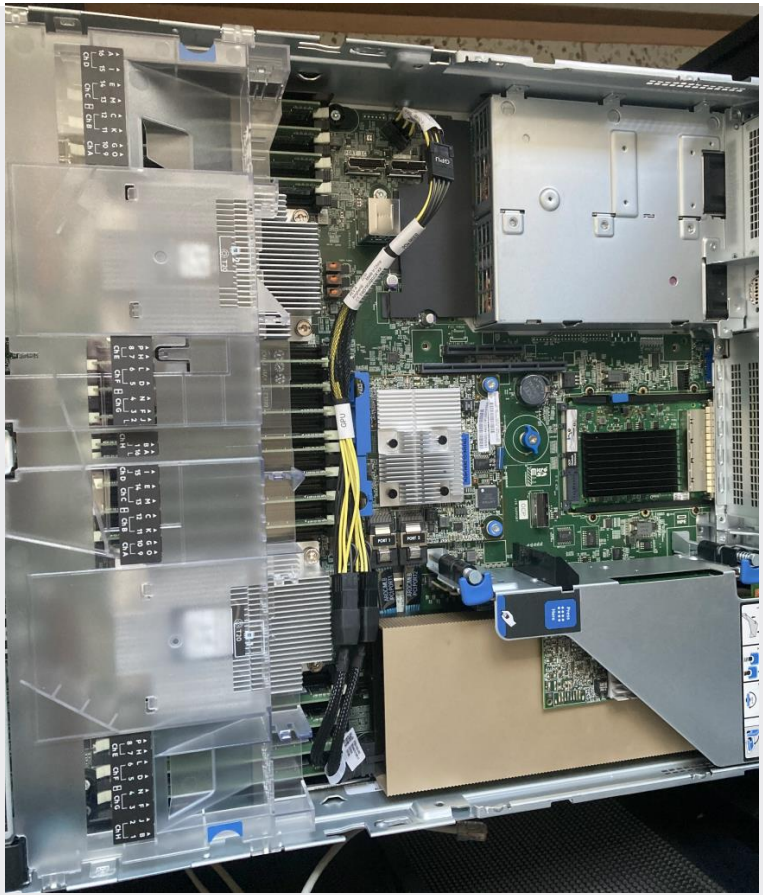
- **NVIDIA GPU - L40**
- **Windows prostředí**
  - Virtualizace – hypervizor Hyper-V

    Windows Server 2019 Datacenter
  - 2-node Cluster Windows Server

Zkušenosti se zprovozněním LLM



Zkušenosti se zprovozněním LLM



Zkušenosti se zprovozněním LLM

- **Windows Server 2022 na Hyper-V (2019)**
- **Pouze single VM (Pass Through - DDA (Discrete Device Assignment))** pouze celá karta jednomu VM

  Win Srv 2025 (preview) – GPU partitioning na více VM
- **Linuxová VM na Hyper-V**
  - NVIDIA Triton Inference Server
  - Ubuntu Server 20.04
  - Nepovedlo se ☹ - závislosti balíků
  - Souborový systém na virtualizovaném disku – nelze „shrinkovat“ stále jen roste velikost (a vždy ji nakonec vyčerpáte)

Problémy, slepé uličky..

- **WSL2**
  - Využívá Hypervisor, takže pokud máme Virtualizovaný server nelze WSL použít.
  - Samostatný HW – ale stejně nejistá podpora pro ovladače karty…
- **Docker**
  - Využívá Hypervizor
  - Nutná vnořená virtualizace – VMware ESXi and Azure

„Nested virtualization is supported by Microsoft for running Hyper-V inside an Azure VM“

Funkční konfigurace

- **HW – HP DL385 G10, NVIDIA GPU - L40**
- **Windows Server**
  - Virtualizace Hyper-V – Host OS: Windows Server 2019 Datacenter
    - 2-node cluster (1 GPU karta, takže zde pouze na 1 node)
    - zde se neinstaluje GPU ovladač a zařízení GPU se z něj v podstatě vyjme.
  - Guest OS (VM) WinSrv 2022 – Pass Through (DDA - Discrete Device Assignment)
    - GPU Driver NVIDIA L40 537.70
    - CUDA 12.1
    - Python 3.11
    - Torch 2.5
    - LM Studio (llama.cpp)

Funkční konfigurace

- **Windows admin GUI** (lze nastavit pouze minimum věcí)
- **Powershell** (Host OS)

- Zakázat GPU

```
PS C:\Temp> $pnpdevs | Where-Object {$_.Class -like "Display" -and $_.Manufacturer -like "NVIDIA"}
PS C:\Temp> $gpudevs = $pnpdevs | Where-Object {$_.Class -like "Display" -and $_.Manufacturer -like "NVIDIA"}
PS C:\Temp>
PS C:\Temp> disable-PnpDevice -InstanceId $gpudevs[0].InstanceId
```

- Vyjmout zařízení

```
PS C:\Temp> # Dismount device
PS C:\Temp> $locationPath = ($gpudevs | Get-PnpDeviceProperty DEVPKEY_Device_LocationPaths).data[0]
PS C:\Temp> Dismount-VMHostAssignableDevice -Force -LocationPath $locationPath
```

Funkční konfigurace

- **Host OS**

- Konfigurace VM, MMIO, přiřazení GPU v režimu Pass-Trough

```
PS C:\Temp> # Configure the VM for a Discrete Device Assignment
PS C:\Temp> $vm = "VMGPU01"
PS C:\Temp>
PS C:\Temp> # Set automatic stop action to TurnOff
PS C:\Temp> Set-VM -Name $vm -AutomaticStopAction TurnOff
PS C:\Temp>
PS C:\Temp> # Enable Write-Combining on the CPU
PS C:\Temp> Set-VM -GuestControlledCacheTypes $true -VMName $vm
PS C:\Temp>
PS C:\Temp> # Configure 32 bit MMIO space
PS C:\Temp> Set-VM -LowMemoryMappedIoSpace 3Gb -VMName $vm
PS C:\Temp>
PS C:\Temp> # Configure Greater than 32 bit MMIO space  (!! L40 GPU is a 48GB card !! --> 65GB)
PS C:\Temp> Set-VM -HighMemoryMappedIoSpace 65Gb -VMName $vm
PS C:\Temp>
PS C:\Temp> Add-VMAssignableDevice -LocationPath $locationPath -VMName $vm
```

- **Start VM**

Funkční konfigurace

- **Host OS**

- Pokud budete GPU ze serveru vyjímat ven, nutno nejprve odebrat z VM !!!

```
PS C:\Temp> # Remove the device from the VM
PS C:\Temp> Remove-VMAssignableDevice -LocationPath $locationPath -VMName VMGPU01
PS C:\Temp>
PS C:\Temp> # Mount the device back in the host
PS C:\Temp> Mount-VMHostAssignableDevice -LocationPath $locationPath
```

Funkční konfigurace

- **Guest OS (Windows Server 2022 Standard** s Desktop Experience**)**
  - Vzor konfigurace: google colab (https://colab.google)
  - GPU Driver – NVIDIA_L40_Windows_Driver_537.70.exe
  - CUDA 12.1 – cuda_12.1.0_531.14_windows.exe
  - Python 3.11 – python-3.11.9-amd64.exe
  - Torch 2.5
  - LM Studio (llama.cpp)

Problémy a postřehy

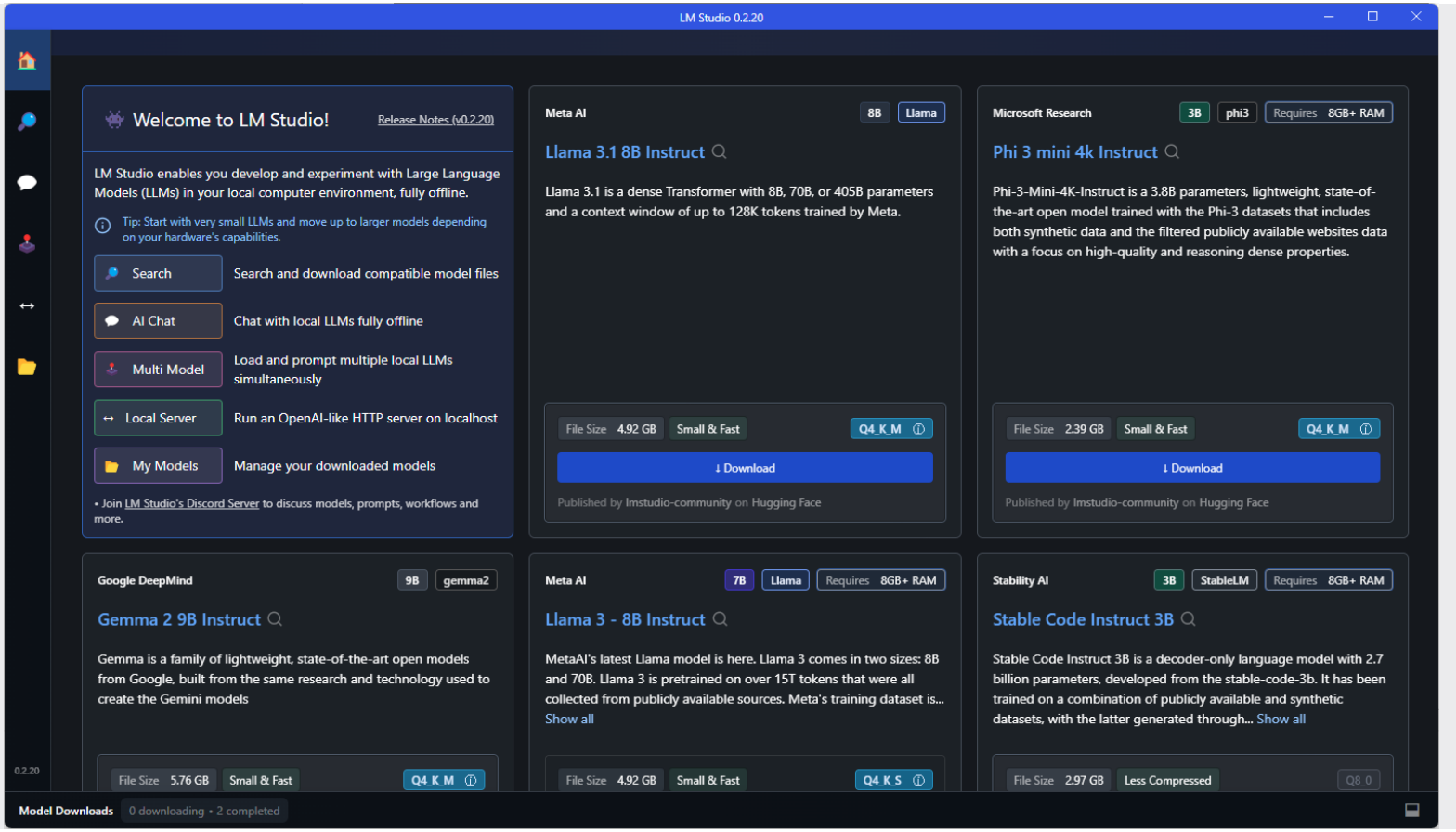
Postřehy

- **GPU se bez potíží podařilo nainstalovat do serveru** a **bez větších potíží** ji **rozběhnout** a zpřístupnit **ve** virtualizovaném prostředí Hyper-V příslušné **VM.**
- **Ovladače karty a rozhraní CUDA** se ve Windows 2022 podařilo rozběhnout **bez větších potíží**.

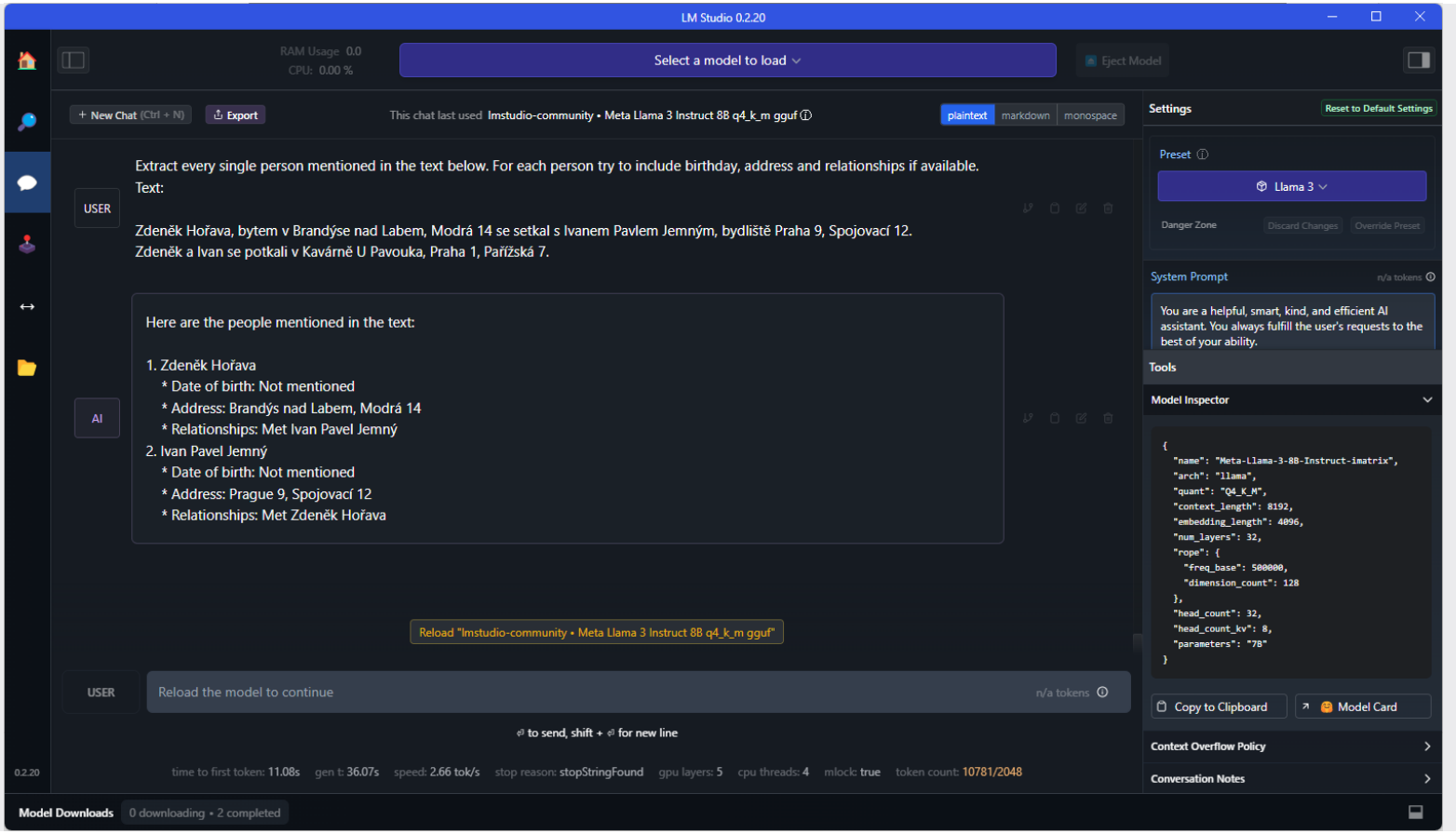
Problémy

- **Nelze použít standardní Performance countery Windows**. Informace o utilizaci GPU a VRAM byly k dispozici **pouze** v **Control panelu NVIDIA** a z logů.
- **Byl učiněn pokus (několikrát) o rozběhnutí NVIDIA Triton Inference Server** (virtuální server s linuxovou distribucí Ubuntu Server 22.04). Server se podle návodů v linuxové VM nepodařilo zprovoznit. Vždy to **skončilo neúspěchem** ve fázi sestavení po konverzi checkpointů huggingface.co modelu, s chybou na nějaké závislosti python knihoven. Byly zkoušeny různé kombinace verzí CUDA a TensorRT-LLM a modelů LLAMA, ale vždy to skončilo neúspěchem. Protože se vždy jednalo o časově náročné pokusy, vzdali jsme tuto snahu a věnovali se výhradně testování na Windows platformě.

LM Studio


LM Studio 0.2.20
Welcome to LM Studio!
Release Notes (v0.2.20)
LM Studio enables you develop and experiment with Large Language Models (LLMs) in your local computer environment, fully offline.
Tip: Start with very small LLMs and move up to larger models depending on your hardware's capabilities.
Search
Search and download compatible model files
AI Chat
Chat with local LLMs fully offline
Multi Model
Load and prompt multiple local LLMs simultaneously
Local Server
Run an OpenAI-like HTTP server on localhost
My Models
Manage your downloaded models
• Join LM Studio's Discord Server to discuss models, prompts, workflows and more.
Meta AI
8B
Llama
Llama 3.1 8B Instruct
Llama 3.1 is a dense Transformer with 8B, 70B, or 405B parameters and a context window of up to 128K tokens trained by Meta.
File Size 4.92 GB
Small & Fast
Q4_K_M
↓ Download
Published by lmstudio-community on Hugging Face
Microsoft Research
3B
phi3
Requires 8GB+ RAM
Phi 3 mini 4k Instruct
Phi-3-Mini-4K-Instruct is a 3.8B parameters, lightweight, state-of-the-art open model trained with the Phi-3 datasets that includes both synthetic data and the filtered publicly available websites data with a focus on high-quality and reasoning dense properties.
File Size 2.39 GB
Small & Fast
Q4_K_M
↓ Download
Published by lmstudio-community on Hugging Face
Google DeepMind
9B
gemma2
Gemma 2 9B Instruct
Gemma is a family of lightweight, state-of-the-art open models from Google, built from the same research and technology used to create the Gemini models
File Size 5.76 GB
Small & Fast
Q4_K_M
Meta AI
7B
Llama
Requires 8GB+ RAM
Llama 3 - 8B Instruct
MetaAI's latest Llama model is here. Llama 3 comes in two sizes: 8B and 70B. Llama 3 is pretrained on over 15T tokens that were all collected from publicly available sources. Meta's training dataset is...
Show all
File Size 4.92 GB
Small & Fast
Q4_K_S
Stability AI
3B
StableLM
Requires 8GB+ RAM
Stable Code Instruct 3B
Stable Code Instruct 3B is a decoder-only language model with 2.7 billion parameters, developed from the stable-code-3b. It has been trained on a combination of publicly available and synthetic datasets, with the latter generated through... Show all
File Size 2.97 GB
Less Compressed
Q8_0
0.2.20
Model Downloads
0 downloading • 2 completed

LM Studio


LM Studio 0.2.20
RAM Usage 0.0
CPU: 0.00 %
Select a model to load
Eject Model
+ New Chat (Ctrl + N)
Export
This chat last used lmstudio-community • Meta Llama 3 Instruct 8B q4_k_m gguf
plaintext
markdown
monospace
Settings
Reset to Default Settings
Preset
Llama 3
Danger Zone
Discard Changes
Override Preset
System Prompt
n/a tokens
You are a helpful, smart, kind, and efficient AI assistant. You always fulfill the user's requests to the best of your ability.
Tools
Model Inspector
USER
Extract every single person mentioned in the text below. For each person try to include birthday, address and relationships if available.
Text:
Zdeněk Hořava, bytem v Brandýse nad Labem, Modrá 14 se setkal s Ivanem Pavlem Jemným, bydliště Praha 9, Spojovací 12.
Zdeněk a Ivan se potkali v Kavárně U Pavouka, Praha 1, Pařížská 7.
AI
Here are the people mentioned in the text:
1. Zdeněk Hořava
* Date of birth: Not mentioned
* Address: Brandýs nad Labem, Modrá 14
* Relationships: Met Ivan Pavel Jemný
2. Ivan Pavel Jemný
* Date of birth: Not mentioned
* Address: Prague 9, Spojovací 12
* Relationships: Met Zdeněk Hořava
{
"name": "Meta-Llama-3-8B-Instruct-imatrix",
"arch": "llama",
"quant": "Q4_K_M",
"context_length": 8192,
"embedding_length": 4096,
"num_layers": 32,
"rope": {
"freq_base": 500000,
"dimension_count": 128
},
"head_count": 32,
"head_count_kv": 8,
"parameters": "7B"
}
Reload "lmstudio-community • Meta Llama 3 Instruct 8B q4_k_m gguf"
USER
Reload the model to continue
n/a tokens
to send, shift + ⏎ for new line
Copy to Clipboard
Model Card
Context Overflow Policy
Conversation Notes
0.2.20
time to first token: 11.08s gen t: 36.07s speed: 2.66 tok/s stop reason: stopStringFound gpu layers: 5 cpu threads: 4 mlock: true token count: 10781/2048
Model Downloads
0 downloading • 2 completed

Úlohy testované s použitím LLM

- **Extrakce entit (Python/Torch)**
  - **LLM Inference (meta-llama/Meta-llama-3-8B-Instruct)**
    - Vstup: prostý text
    - Výstup: strukturovaný seznam osob, adres…

- model_id: meta-llama/Meta-Llama-3-8B-Instruct
- number of parameters: 8B
- test data: chunked text file (size 57435B)
- chunk_size: 1536
- chunk_overlap: 50
- platform: Python + Torch 2.5.0
- RAM usage: ~1.2GB
- CPU usage: 1 core, 100%
- **elapsed time: 618 sec.**

Úlohy testované s použitím LLM

Extrakce entit (Python/LM Studio Server)

- **LLM Inference (meta-llama/Meta-llama-3-8B-Instruct)**
  - Vstup: prostý text
  - Výstup: strukturovaný seznam osob, adres…

- model_id: meta-llama/Meta-Llama-3-8B-Instruct
- number of parameters: 8B
- test data: chunked text file (size 57435B)
- chunk_size: 1536
- chunk_overlap: 50
- platform: Python + LM Studio Server
- **elapsed time: 172 sec.**

Úlohy testované s použitím LLM

- **Retrieve-Augment-Generate (RAG)**
  - **Text embeddings (BAAI/bge-m3)**
  - **LLM Inference (meta-llama/Meta-llama-3-8B-Instruct)**
    - Vstup: Sada textových dokumentů, Uživatelský dotaz v prostém jazyce
    - Výstup: odpověď v přirozeném jazyce

- model_id - embeddings: BAAI/bge-m3
- model_id - generation: meta-llama/Meta-Llama-3-8B-Instruct (gguf version)
- test data: chunked text documents
- chunk_size: 1024
- chunk_overlap: 100
- platform - embeddings: Python + Torch 2.5.0
- platform - generation: Python + LMStudio Server
- **elapsed time – get embedding vector of question string: ~0.03s**
- **elapsed time - generation: 8-10s**

Úlohy testované s použitím LLM

- **Transformace vyhledávacích dotazů**
  - **LLM Fine-Tuning (microsoft/Phi-3-mini-4k-instruct)**
  - **LLM Inference**
    - Vstup: dotaz v přirozeném jazyce
    - Výstup: dotaz ve strukturované syntaxi

VYHLEDÁVÁNÍ PŘIROZENÝM JAZYKEM (Text-to-SxQL)

Funkce, která převede volný text zapsaný uživatelem na text ve vyhledávací syntaxi Sx (SxQL)

- **Vstup**: volný text, který popisuje hledané subjekty
- **Výstup**: text v SxQL syntaxi

Příklady – reálný vstup/výstup:

```
Uživatel: najdi osoby věk 45-50 s vazbou na organizaci Pilsner Urguell
SxQL     : TypObj:(OSO) Věk: (45~50) @joinBy @joinTo TypObj:(ORG) Pilsner Urguell
```

Děkuji za pozornost