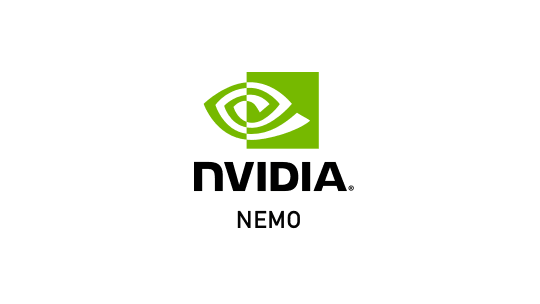
Source: https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/nemotron-4-340b-base

[Models](https://catalog.ngc.nvidia.com/models)

Nemotron-4-340B-Base

**Nemotron-4-340B-Base**

Download



Description

Nemotron-4-340B-Base

Publisher

NVIDIA

Latest Version

1.2

Modified

November 12, 2024

Size

635.23 GB

Conversational AINSPECT-CAZM-5YF1Natural Language ProcessingNatural Language UnderstandingNeMoSynthetic Data Generationnvidia

OverviewModel Card++Version HistoryFile BrowserRelated Collections

**Nemotron-4-340B-Base**

**Model Overview:**

Nemotron-4-340B-Base is a large language model (LLM) that can be used as part of a synthetic data generation pipeline to create training data that helps researchers and developers build their own LLMs. This model has 340 billion parameters, and supports a context length of 4,096 tokens. It is pre-trained for a total of 9 trillion tokens, consisting of a diverse assortment of English-based texts, 50+ natural languages and 40+ coding languages. After an initial pre-training phase of 8 trillion tokens, continuous pre-training of 1 trillion tokens was performed on top of the pre-trained model in order to improve model quality. During continuous pre-training, we altered the data composition by using a different distribution than the one seen at the beginning of training.

Under the NVIDIA Open Model License, NVIDIA confirms: Models are commercially usable. You are free to create and distribute Derivative Models. NVIDIA does not claim ownership to any outputs generated using the Models or Derivative Models

**License:**

[NVIDIA Open Model License](https://developer.download.nvidia.com/licenses/nvidia-open-model-license-agreement-june-2024.pdf)

**Intended use**

Nemotron-4-340B-Base is a completion model intended for use in over 50+ natural and 40+ coding languages. It is compatible with [NVIDIA NeMo Framework](https://docs.nvidia.com/nemo-framework/index.html). For best performance on a given task, users are encouraged to customize the model using the NeMo Framework suite of customization tools including Parameter-Efficient Fine-Tuning (P-tuning, Adapters, LoRA, and more), and Model Alignment (SFT, SteerLM, RLHF, and more) using [NeMo-Aligner](https://github.com/NVIDIA/NeMo-Aligner).

**Model Developer:** NVIDIA

**Model Dates:** Nemotron-4-340B-Base was trained between December 2023 and May 2024.

**Required Hardware**

BF16 Inference:

* 8x H200 (1x H200 node)
* 16x H100 (2x H100 nodes)
* 16x A100 80GB (2x A100 80GB nodes)

**Model Architecture:**

Nemotron-4-340B-Base is trained with a global batch-size of 2304, a sequence length of 4096 tokens, uses Grouped-Query Attention (GQA), and Rotary Position Embeddings (RoPE).

**Architecture Type:** Transformer Decoder (auto-regressive language model)

**Network Architecture:** Nemotron-4

**Usage**

First unpack the .nemo file with:

mkdir Nemotron-4-340B-Base && cd Nemotron-4-340B-Base && tar xf /path/to/nemotron-4-340b-base-v1.nemo

Deployment and inference with Nemotron-4-340B-Base can be done in three steps using NeMo Framework:

Create a Python script to interact with the deployed model. Create a Bash script to start the inference server. Schedule a Slurm job to distribute the model across 2 nodes and associate them with the inference server.

1. Define the Python script call\_server.py

import requests

import json

headers = {"Content-Type": "application/json"}

def text\_generation(data, ip='localhost', port=None):

resp = requests.put(f'http://{ip}:{port}/generate', data=json.dumps(data), headers=headers)

return resp.json()

def get\_generation(prompt, greedy, add\_BOS, token\_to\_gen, min\_tokens, temp, top\_p, top\_k, repetition, batch=False):

data = {

"sentences": [prompt] if not batch else prompt,

"tokens\_to\_generate": int(token\_to\_gen),

"temperature": temp,

"add\_BOS": add\_BOS,

"top\_k": top\_k,

"top\_p": top\_p,

"greedy": greedy,

"all\_probs": False,

"repetition\_penalty": repetition,

"min\_tokens\_to\_generate": int(min\_tokens),

"end\_strings": ["<|endoftext|>", "<extra\_id\_1>", "\x11", "<extra\_id\_1>User"],

}

sentences = text\_generation(data, port=1424)['sentences']

return sentences[0] if not batch else sentences

PROMPT\_TEMPLATE = "{prompt}"

question = "Write a poem on NVIDIA in the style of Shakespeare"

prompt = PROMPT\_TEMPLATE.format(prompt=question)

print(prompt)

response = get\_generation(prompt, greedy=True, add\_BOS=False, token\_to\_gen=1024, min\_tokens=1, temp=1.0, top\_p=1.0, top\_k=0, repetition=1.0, batch=False)

response = response[len(prompt):]

print(response)

1. Given this Python script, create a Bash script which spins up the inference server within the [NeMo container](https://catalog.ngc.nvidia.com/orgs/nvidia/containers/nemo) (docker pull nvcr.io/nvidia/nemo:24.01.framework) and calls the Python script call\_server.py. The Bash script nemo\_inference.sh is as follows:

NEMO\_FILE=$1

WEB\_PORT=1424

depends\_on () {

HOST=$1

PORT=$2

STATUS=$(curl -X PUT http://$HOST:$PORT >/dev/null 2>/dev/null; echo $?)

while [ $STATUS -ne 0 ]

do

echo "waiting for server ($HOST:$PORT) to be up"

sleep 10

STATUS=$(curl -X PUT http://$HOST:$PORT >/dev/null 2>/dev/null; echo $?)

done

echo "server ($HOST:$PORT) is up running"

}

/usr/bin/python3 /opt/NeMo/examples/nlp/language\_modeling/megatron\_gpt\_eval.py \

gpt\_model\_file=$NEMO\_FILE \

pipeline\_model\_parallel\_split\_rank=0 \

server=True tensor\_model\_parallel\_size=8 \

trainer.precision=bf16 pipeline\_model\_parallel\_size=2 \

trainer.devices=8 \

trainer.num\_nodes=2 \

web\_server=False \

port=${WEB\_PORT} &

SERVER\_PID=$!

readonly local\_rank="${LOCAL\_RANK:=${SLURM\_LOCALID:=${OMPI\_COMM\_WORLD\_LOCAL\_RANK:-}}}"

if [ $SLURM\_NODEID -eq 0 ] && [ $local\_rank -eq 0 ]; then

depends\_on "0.0.0.0" ${WEB\_PORT}

echo "start get json"

sleep 5

echo "SLURM\_NODEID: $SLURM\_NODEID"

echo "local\_rank: $local\_rank"

/usr/bin/python3 /scripts/call\_server.py

echo "clean up dameons: $$"

kill -9 $SERVER\_PID

pkill python

fi

wait

1. Launch nemo\_inference.sh with a Slurm script defined like below, which starts a 2-node job for model inference.

#!/bin/bash

#SBATCH -A SLURM-ACCOUNT

#SBATCH -p SLURM-PARITION

#SBATCH -N 2

#SBATCH -J generation

#SBATCH --ntasks-per-node=8

#SBATCH --gpus-per-node=8

set -x

RESULTS=<PATH\_TO\_YOUR\_SCRIPTS\_FOLDER>

OUTFILE="${RESULTS}/slurm-%j-%n.out"

ERRFILE="${RESULTS}/error-%j-%n.out"

MODEL=<PATH\_TO>/Nemotron-4-340B-Base

MOUNTS="--container-mounts=<PATH\_TO\_YOUR\_SCRIPTS\_FOLDER>:/scripts,MODEL:/model"

read -r -d '' cmd <<EOF

bash /scripts/nemo\_inference.sh /model

EOF

srun -o $OUTFILE -e $ERRFILE --container-image="$CONTAINER" $MOUNTS bash -c "${cmd}"

**Dataset & Training**

The training corpus for Nemotron-4-340B-Base consists of English and multilingual text, as well as code. Our sources cover a variety of document types such as: webpages, dialogue, articles, and other written materials. The corpus spans domains including legal, math, science, finance, and more. In our continued training set, we introduce a small portion of question-answering, and alignment style data to improve model performance.

**Data Freshness:** The pretraining data has a cutoff of June 2023.

**Evaluation Results**

**Overview**

*5-shot performance.* Language Understanding evaluated using [Massive Multitask Language Understanding](https://arxiv.org/abs/2009.03300):

| **Average** |
| --- |
| 81.1 |

*Zero-shot performance.* Evaluated using select datasets from the [LM Evaluation Harness](https://github.com/EleutherAI/lm-evaluation-harness) with additions:

| **HellaSwag** | **Winogrande** | **BBH** | **ARC-Challenge** |
| --- | --- | --- | --- |
| 90.53 | 89.50 | 85.44 | 94.28 |

*Chain of Thought (CoT)*. Multilingual capabilities evaluated using [Multilingual Grade School Math](https://arxiv.org/abs/2210.03057):

| **ES Exact Match (%)** | **JA Exact Match (%)** | **TH Exact Match (%)** |
| --- | --- | --- |
| 68.8 | 69.6 | 68.4 |

*Code generation performance*. Evaluated using [HumanEval](https://github.com/openai/human-eval):

| **p@1, 0-Shot** |
| --- |
| 57.3 |

**Limitations**

The model was trained on data that contains toxic language, unsafe content, and societal biases originally crawled from the internet. Therefore, the model may amplify those biases and return toxic responses especially when prompted with toxic prompts. The model may generate answers that may be inaccurate, omit key information, or include irrelevant or redundant text producing socially unacceptable or undesirable text, even if the prompt itself does not include anything explicitly offensive.

**Ethical Considerations**

NVIDIA believes Trustworthy AI is a shared responsibility and we have established policies and practices to enable development for a wide array of AI applications. When downloaded or used in accordance with our terms of service, developers should work with their internal model team to ensure this model meets requirements for the relevant industry and use case and addresses unforeseen product misuse. For more detailed information on ethical considerations for this model, please see the Model Card++ Explainability, Bias, Safety & Security, and Privacy Subcards [here.](https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/nemotron-4-340b-base). Please report security vulnerabilities or NVIDIA AI Concerns [here](https://www.nvidia.com/en-us/support/submit-security-vulnerability/).