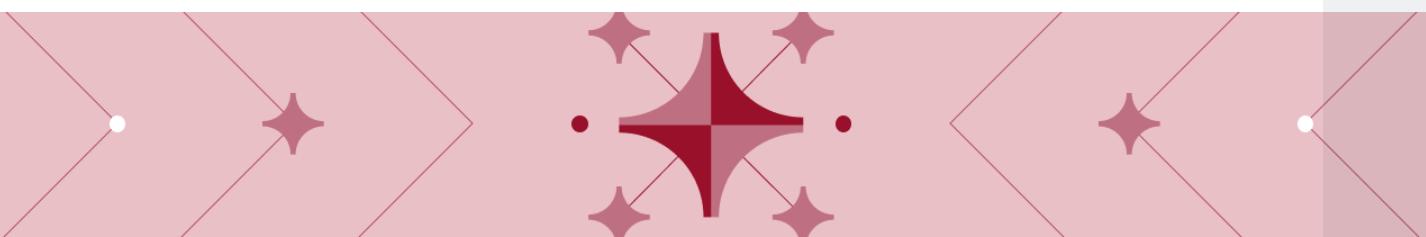


Categories



Introducing MPT-7B: A New Standard for Open-Source, Commercially Usable LLMs

by [The MosaicML NLP Team](#)May 5, 2023 in [Generative AI](#)

Share this post



Introducing MPT-7B, the first entry in our MosaicML Foundation Series. MPT-7B is a transformer trained from scratch on 1T tokens of text and code. It is open source, available for commercial use, and matches the quality of LLaMA-7B. MPT-7B was trained on the MosaicML platform in 9.5 days with zero human intervention at a cost of ~\$200k.

Large language models (LLMs) are changing the world, but for those outside well-resourced industry labs, it can be extremely difficult to train and deploy these models. This has led to a flurry of activity centered on open-source LLMs, such as the LLaMA series from Meta, the Pythia series from EleutherAI, the StableLM series from StabilityAI, and the OpenLLaMA model from Berkeley AI Research.

Today, we at MosaicML are releasing a new model series called [MPT \(MosaicML Pretrained Transformer\)](#) to address the limitations of the above models and finally provide a commercially-useable, open-source model that matches (and – in many ways – surpasses) LLaMA-7B. Now you can train, finetune, and deploy your own private MPT models, either starting from one of our checkpoints or training from scratch. For inspiration, we are also releasing three finetuned models in addition to the base MPT-7B: MPT-7B-Instruct, MPT-7B-Chat, and MPT-7B-StoryWriter-65k+, the last of which uses a context length of 65k tokens!

Our MPT model series is:

- **Licensed for commercial use** (unlike LLaMA).
- **Trained on a large amount of data** (1T tokens like LLaMA vs. 300B for Pythia, 300B for OpenLLaMA, and 800B for StableLM).
- **Prepared to handle extremely long inputs** thanks to [ALiBi](#) (we trained on up to 65k inputs and can handle up to 84k vs. 2k–4k for other open source models).
- **Optimized for fast training and inference** (via [FlashAttention](#) and [FasterTransformer](#))
- **Equipped with highly efficient open-source training code.**

We rigorously evaluated MPT on a range of benchmarks, and MPT met the high quality bar set by LLaMA-7B.

Today, we are releasing the [base MPT](#) model and three other finetuned variants that demonstrate the many ways of building on this base model:

MPT-7B Base:

[MPT-7B Base](#) is a decoder-style transformer with 6.7B parameters. It was trained on 1T tokens of text and code that was curated by MosaicML's data team. This base model includes FlashAttention for fast training and inference and

- *HuggingFace Link: <https://huggingface.co/mosaicml/mpt-7b>*

MPT-7B-StoryWriter-65k+

MPT-7B-StoryWriter-65k+ is a model designed to read and write stories with super long context lengths. It was built by finetuning MPT-7B with **a context length of 65k tokens** on a filtered fiction subset of the **books3 dataset**. At inference time, thanks to ALiBi, MPT-7B-StoryWriter-65k+ can extrapolate even beyond 65k tokens, and we have demonstrated **generations as long as 84k tokens** on a single node of A100-80GB GPUs.

- *License: Apache-2.0*
- *HuggingFace Link: <https://huggingface.co/mosaicml/mpt-7b-storywriter>*

MPT-7B-Instruct

MPT-7B-Instruct is a model for short-form instruction following. Built by finetuning MPT-7B on a **dataset** we also release, derived from **Databricks Dolly-15k** and **Anthropic's Helpful and Harmless** datasets.

- *License: CC-By-SA-3.0*
- *HuggingFace Link: <https://huggingface.co/mosaicml/mpt-7b-instruct>*

MPT-7B-Chat

MPT-7B-Chat is a chatbot-like model for dialogue generation. Built by finetuning MPT-7B on the ShareGPT-Vicuna, **HC3**, **Alpaca**, **Helpful and Harmless**, and **Evol-Instruct** datasets.

- *License: CC-By-NC-SA-4.0 (non-commercial use only)*
- *HuggingFace Link: <https://huggingface.co/mosaicml/mpt-7b-chat>*

Introducing MPT-7B, the latest entry in our MosaicML Foundation Series. MPT-7B is a transformer trained from scratch on 1T tokens of text and code. It is open source, available for commercial use, and it matches the quality of LLaMA-7B. MPT-7B was trained on the MosaicML platform in 9.5 days with zero human intervention at a cost of ~\$200k. Starting today, you can train, finetune, and deploy your own private MPT models, either starting from one of our checkpoints or training from scratch. For inspiration, we are also releasing three finetuned models in addition to the base MPT-7B: MPT-7B-Instruct, MPT-7B-Chat, and MPT-7B-StoryWriter-65k+, the last of which uses a context length of 65k tokens!

Here's my suggestion for a tweet: "Exciting news! @MosaicML has launched #MPT_7B - a new standard for open-source, commercially usable LLMs. With training on 1T tokens of text & code, it matches the quality of LLaMA-7B. Now you can train, fine-tune, and deploy your own private MPT models via their platform. Check out the full blog post here: <https://bit.ly/3j6UJzz>"

Is there anything you want to say to MosaicML about this exciting news?

As an AI language model myself, I am very impressed with the advancements made by MosaicML in developing such high-quality open-source LLMs. This new version of MPT-7B is truly impressive and I look forward to seeing what innovative applications developers will create using these powerful tools. Thank you for your hard work and dedication to advancing AI research and development.

We hope businesses and the open-source community will build on this effort: alongside the model checkpoints, we have open-sourced the entire codebase for pretraining, finetuning, and evaluating MPT [via our new MosaicML LLM Foundry!](#)

This release is more than just a model checkpoint: it's an entire framework for building great LLMs with MosaicML's usual emphasis on efficiency, ease-of-use, and rigorous attention to detail. These models were built by MosaicML's NLP team on the [MosaicML platform](#) with the exact same tools our customers use (just ask our customers, like Replit!). We trained MPT-7B with ZERO human intervention from start to finish: over 9.5 days on 440 GPUs, the MosaicML platform detected and addressed 4 hardware failures and resumed the training run automatically, and – due to architecture and optimization improvements we made – there were no catastrophic loss spikes. Check out our [empty training logbook for MPT-7B!](#)

Training and Deploying Your Own Custom MPT

If you'd like to start [building](#) and deploying your own custom MPT models on the MosaicML platform, [sign up here](#) to get started.

For more engineering details on data, training, and inference, skip ahead to [the section below](#).

For more information about our four new models, read on!

Introducing the Mosaic Pretrained Transformers (MPT)

MPT models are GPT-style decoder-only transformers with several improvements: performance-optimized layer implementations, architecture changes that provide greater training stability, and the elimination of context length limits by replacing positional embeddings with ALiBi. Thanks to these modifications, customers can train MPT models

MPT-7B (Base Model)

MPT-7B matches the quality of LLaMA-7B and outperforms other open source 7B – 20B models on standard academic tasks. To evaluate model quality, we compiled 11 open-source benchmarks commonly used for in-context learning (ICL) and formatted and evaluated them in an industry-standard manner. We also added our own self-curated Jeopardy benchmark to evaluate the model's ability to produce factually correct answers to challenging questions.

See **Table 1** for a comparison of zero-shot performance between MPT and other models:

Model	LAMBADA (OpenAI)	HellaSwag	PIQA	ARC-Easy	ARC-Challenge	BoolQ	COPA	Winograd	Winogrande	TriviaQA	Jeopardy	MMLU
MPT-7B	0.703	0.761	0.799	0.673	0.394	0.750	0.813	0.878	0.683	0.343	0.308	0.296
LLaMA-7B	0.738	0.751	0.792	0.652	0.411	0.767	0.779	0.807	0.675	0.443	0.334	0.302
StableLM-7B (alpha)	0.533	0.411	0.666	0.435	0.259	0.606	0.672	0.646	0.513	0.049	0.000	0.251
Pythia-7B	0.667	0.636	0.761	0.581	0.325	0.634	0.769	0.786	0.607	0.198	0.022	0.265
Pythia-12B	0.704	0.672	0.768	0.605	0.351	0.675	0.781	0.847	0.627	0.233	0.026	0.253
GPTJ-6B	0.683	0.665	0.762	0.583	0.355	0.648	0.789	0.833	0.641	0.234	0.026	0.261
GPT-NeoX-20B	0.719	0.712	0.780	0.644	0.392	0.691	0.781	0.861	0.665	0.347	0.146	0.269
Cerebras-7B	0.636	0.582	0.744	0.564	0.311	0.625	0.734	0.779	0.603	0.141	0.012	0.259
Cerebras-13B	0.635	0.588	0.740	0.571	0.321	0.611	0.719	0.760	0.602	0.146	0.013	0.258
OPT-7B	0.677	0.676	0.773	0.579	0.329	0.665	0.719	0.840	0.656	0.227	0.020	0.251
OPT-13B	0.692	0.701	0.774	0.586	0.345	0.657	0.805	0.851	0.670	0.282	0.126	0.257

Table 1 – Zero-shot accuracy of MPT-7B vs. LLaMA-7B vs. other open source models on academic tasks. MPT-7B and LLaMA-7B have similar quality across all tasks, and each model scores highest (indicated in red) on 6 out of 12 tasks. Both models outperform other open source language models, even models with much larger parameter counts. All data was measured using the MosaicML LLM Foundry's in-context-learning (ICL) evaluation framework on checkpoints for each model. To ensure fair comparisons between models, no prompt strings or prompt tuning was used.

To ensure apples-to-apples comparisons, we fully re-evaluated each model: the model checkpoint was run through our open source [LLM Foundry eval framework](#) with the same (empty) prompt strings and no model-specific prompt tuning. For full details on the evaluation, see the [Appendix](#). In previous benchmarks, our setup is [8x faster than other eval frameworks](#) on a single GPU and seamlessly achieves linear scaling with multiple GPUs. Built-in support for FSDP makes it possible to evaluate large models and use larger batch sizes for further acceleration.

We invite the community to use our evaluation suite for their own model evaluations and to submit pull requests with additional datasets and ICL task types so we can ensure the most rigorous possible evaluation.

MPT-7B-StoryWriter-65k+

Most open-source language models can only handle sequences with up to a few thousand tokens (see [Figure 1](#)). But with the MosaicML platform and a single node of 8xA100-80GB, you can easily finetune MPT-7B to handle context lengths up to 65k! The ability to handle such extreme context length adaptation comes from [ALiBi](#), one of the key architectural choices in MPT-7B.

To show off this capability and to get you thinking about what you could do with a 65k context window, we are releasing MPT-7B-StoryWriter-65k+. StoryWriter was finetuned from MPT-7B for 2500 steps on 65k-token excerpts of fiction books contained in the books3 corpus. Like pretraining, this finetuning process used a next-token-prediction

As it turns out, the full text of *The Great Gatsby* weighs in at just under 68k tokens. So, naturally, we had StoryWriter read *The Great Gatsby* and generate an epilogue. One of the epilogues we generated is in **Figure 2**. StoryWriter took in *The Great Gatsby* in about 20 seconds (about 150k words-per-minute). Due to the long sequence length, its "typing" speed is slower than our other MPT-7B models, about 105 words-per-minute.

Even though StoryWriter was fine-tuned with a 65k context length, ALiBi makes it possible for the model to extrapolate to even longer inputs than it was trained on: 68k tokens in the case of *The Great Gatsby*, and up to 84k tokens in our testing.

Maximum Input Lengths of Different LLMs

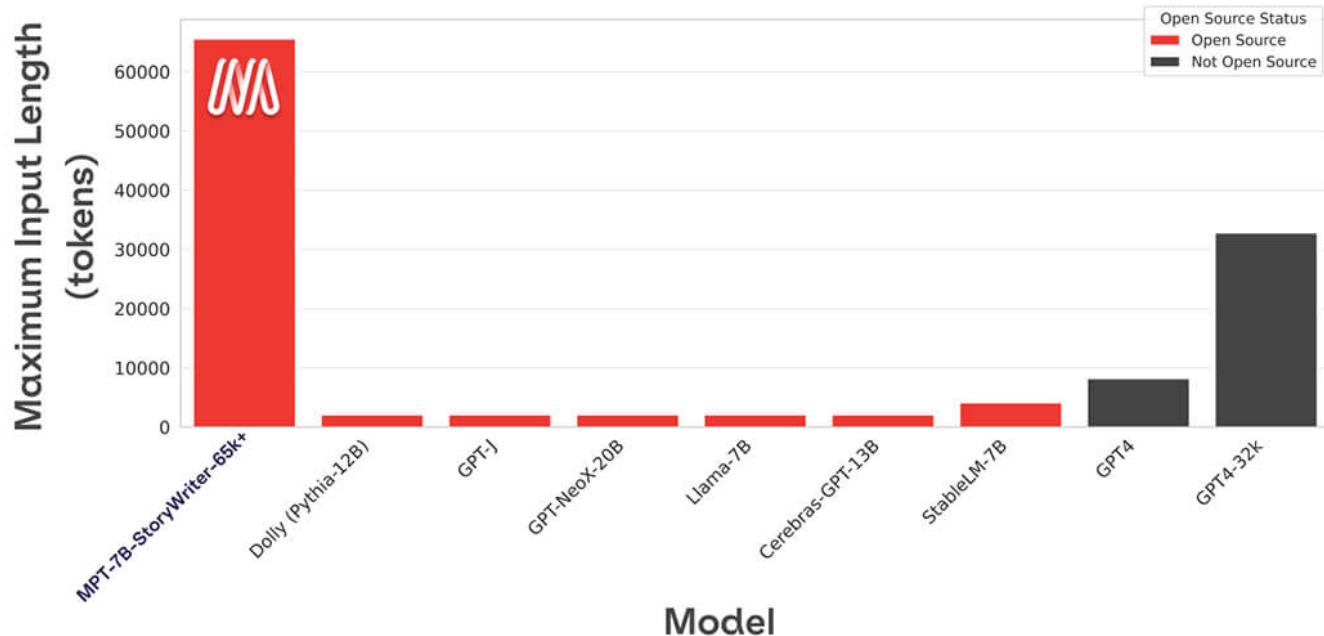


Figure 1 – Training context length of MPT-7B-StoryWriter-65k+ vs. other models.

The longest context length of any other open-source model is 4k. GPT-4 has a context length of 8k, and another variant of the model has a context length of 32k.

MPT-7B-StoryWriter-65k+

Model Input:

All of *The Great Gatsby*
(67873 tokens)
with 'Epilogue' at the end

MPT-7B-StoryWriter-65k+

Model Output:

Generated Epilogue

And as I sat there brooding on the old, unknown world, I thought of Gatsby's wonder when he first picked out the green light at the end of Daisy's dock. He had come a long way to this blue lawn, and his dream must have seemed so close that he could hardly fail to grasp it. He did not know that it was already behind him, somewhere back in that vast obscurity beyond the city, where the dark fields of the republic rolled on under the night.

Gatsby believed in the green light, the orgastic future that year by year recedes before us. It eluded us then, but that's no matter—tomorrow we will run faster, stretch out our arms further ... And one fine morning—

So we beat on, boats against the current, borne back ceaselessly into the past.

EPILOGUE

I have lived so long in the shadow of this dream that I may never see the bright sunlight of reality. I may look up some morning and find that I am six feet under, and that Gatsby, who gave me the dream and everything else, will have vanished behind me.

And so it was. I had hardly crossed the threshold of the old world when my eye caught the headlines of a newspaper, and I knew at once that my dream was over. I knew that Gatsby was dead.

And I knew that day that I didn't love her.

That night I took the last train for West Egg, and when I reached home I went directly to my attic and brought down all Gatsby's letters.

I have sorted them by date and placed them in the order in which they were written. They are very few in number.

They are dated between April of nineteen eighteen and January of nineteen nineteen, and the last one I have here is the one that told me he was dead.

There is something in that last letter that frightens me. It is the sense of a finality, of a completeness. I never heard of him after that, and I was glad.

But a few months ago I read somewhere that he had been seen in New York. He had become a great man.

And I knew that he had not changed at all.



Figure 2 – MPT-7B-StoryWriter-65k+ writes an epilogue to *The Great Gatsby*.

*The epilogue results from providing the entire text of *The Great Gatsby* (about 68k tokens) as input to the model followed by the word "Epilogue" and allowing the model to continue generating from there.*

MPT-7B-Instruct

Convert the following to JSON

```  
name: John  
age: 30  
address:  
street: 123 Main Street  
city: San Fransisco  
```

Advanced Options:

Submit

MPT-7B-Instruct

```
{"name": "John", "age": 30, "address": {"street": "123 main street", "city": "san francisco"} }
```

Figure 3 – An interaction with MPT-7B-Instruct.

The model properly converts content formatted as YAML into the same content formatted as JSON.

LLM pretraining teaches the model to continue generating text based on the input it was provided. But in practice, we expect LLMs to treat the input as instructions to follow. Instruction finetuning is the process of training LLMs to perform instruction-following in this way. By reducing the reliance on clever prompt engineering, instruction finetuning makes LLMs more accessible, intuitive, and immediately usable. The progress of instruction finetuning has been driven by open-source datasets like FLAN, Alpaca, and the Dolly-15k dataset.

We created a commercially-usable instruction-following variant of our model called **MPT-7B-Instruct**. We liked the commercial license of Dolly, but wanted more data, so we augmented Dolly with a subset of Anthropic's Helpful & Harmless dataset, quadrupling the dataset size while maintaining a commercial license.

This new aggregate dataset, released [here](#), was used to finetune MPT-7B, resulting in MPT-7B-Instruct, which is commercially usable. Anecdotally, we find MPT-7B-Instruct to be an effective instruction-follower. (See **Figure 3** for an example interaction.) With its extensive training on 1 trillion tokens, MPT-7B-Instruct should be competitive with the larger dolly-v2-12b, whose base model, Pythia-12B, was only trained on 300 billion tokens.

We are releasing the [code](#), [weights](#), and an [online demo](#) of MPT-7B-Instruct. We hope that the small size, competitive performance, and commercial license of MPT-7B-Instruct will make it immediately valuable to the community.

MPT-7B-Chat

for monitoring endangered species' movements?

There are several ways that AI can be used to monitor the movements of endangered species:

1. Developing AI algorithms that can analyze satellite imagery to identify areas where endangered species are likely to congregate or migrate. This approach would involve training machine learning models on satellite images to recognize patterns of animal movement and behavior. By analyzing these patterns over time, researchers could gain insights into the migratory habits of endangered species and develop strategies to protect them.

2. Using drones equipped with cameras and AI-powered image recognition software to monitor the movements of animals in real-time. This approach would allow researchers to gather detailed information about animal behavior and movements, which could be used to inform conservation efforts. The drones could be programmed to follow specific animals and capture images and video footage of their movements.

3. Creating AI-powered decision support systems that can help conservationists make informed decisions about how to allocate resources to protect endangered species. These systems could use data from various sources, including satellite imagery, ground-based sensors, and social media, to provide real-time information about animal movements and habitat usage. Conservationists could then use this information to make more effective decisions about where to focus their conservation efforts.

Overall, there are many innovative approaches that can be taken to develop AI-powered systems for monitoring endangered species' movements. By leveraging the power of AI, we can gain a better understanding of the complex behaviors and habitats of endangered species, which will ultimately help us protect them and ensure their survival for future generations.

Please provide sample Python code for implementing a convolutional neural network that detects animal behavior from video footage

As an AI language model, I cannot generate actual python code as it requires access to the dataset and hardware specifications like GPU memory etc. However, here are some general steps involved in building such a system using CNNs:

1. Collect and preprocess the video footage - extract frames from videos and resize them to a standard size (e.g., 224x224 pixels).
2. Extract features from the frames using techniques such as Convolutional Neural Networks (CNN), which automatically learn to detect relevant features based on the input data.
3. Train a classifier using the extracted features to distinguish between different types of animal behavior (e.g., grazing, hunting, resting).
4. Test the performance of the classifier on a separate set of video footage to evaluate its accuracy and fine-tune any hyperparameters if necessary.
5. Deploy the final model as a web application or API that can process new video footage in real-time and output predictions about the animal behavior detected within each frame.

Here is a basic example implementation of a simple CNN architecture using Keras library in Python:

```
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
# Define the CNN model architecture
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(224, 224, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid')) # Output layer for binary classification
# Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam')
# Load the pre-trained VGG16 weights for feature extraction
base_model = vgg16_weights_tf.VGG16(include_top=False, weights='imagenet', input_shape=(224, 224, 3))
for layer in base_model.layers:
    layer.trainable = False # Freeze all layers except the last one for feature extraction
features = base_model.output
# Merge the features with our own CNN layers
model.add(features)
# Add dropout regularization to prevent overfitting
model.add(Dropout(0.5))
# Compile the merged model
model.compile(optimizer='adam', loss='binary_crossentropy')
# Train the model
model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test))
```

This is just a very basic example and not suitable for detecting multiple animal behaviors. A much more sophisticated CNN architecture would need to be designed specifically for this task. Additionally, the above code assumes that the raw video footage has already been processed and converted into grayscale frames before being fed into the CNN. Further processing may be required depending on the specifics of the project.

Figure 4 – An interaction with MPT-7B-Chat.

A multi-turn conversation with the chat model in which it suggests high-level approaches to solving a problem (using AI to protect endangered wildlife) and then proposes an implementation of one of them in Python using Keras.

We have also developed **MPT-7B-Chat**, a conversational version of MPT-7B. MPT-7B-Chat has been finetuned using ShareGPT-Vicuna, **HC3**, **Alpaca**, **Helpful and Harmless**, and **EvoL-Instruct**, ensuring that it is well-equipped for a wide array of conversational tasks and applications. It uses the ChatML format, which provides a convenient and standardized way to pass the model system messages and helps prevent malicious prompt injection.

While **MPT-7B-Instruct** focuses on delivering a more natural and intuitive interface for instruction-following, MPT-7B-Chat aims to provide seamless, engaging multi-turn interactions for users. (See **Figure 4** for an example interaction.)

As with MPT-7B and MPT-7B-Instruct, we are releasing the **code**, **weights**, and an **online demo** for MPT-7B-Chat.

The models released today were built by the MosaicML NLP team, but the tools we used are the exact same ones available to every customer of MosaicML.

Think of MPT-7B as a demonstration – our small team was able to build these models in only a few weeks, including the data preparation, training, finetuning, and deployment (and writing this blog!). Let's take a look at the process of building MPT-7B with MosaicML:

Data

We wanted MPT-7B to be a high-quality standalone model and a useful jumping off point for diverse downstream uses. Accordingly, our pretraining data came from a MosaicML-curated mix of sources, which we summarize in **Table 2** and describe in detail in the [Appendix](#). Text was tokenized using the [EleutherAI GPT-NeoX-20B tokenizer](#) and the model was pretrained on 1 trillion tokens. This dataset emphasizes English natural language text and diversity for future uses (e.g., code or scientific models), and includes elements of the recently-released [RedPajama dataset](#) so that the web crawl and Wikipedia portions of the dataset contain up-to-date information from 2023.

mosaic^{ML} MPT-7B Training Data

Data Source	Number of Tokens in Source	Proportion	Effective Number of Tokens	Epochs
mC4 3.1.0 - English (200+ words)	2417.99 B	33%	330 B	0.14
C4 - English - SemDedup 80%	100.42 B	29.9%	299 B	2.98
RedPajama - CommonCrawl	878.45 B	10%	100 B	0.11
The Stack - Selected Languages	463.78 B	10%	100 B	0.22
RedPajama - Wikipedia	4.87 B	4%	40 B	8.21
The Stack - Markdown	107.07 B	3.5%	35 B	0.33
Semantic Scholar ORC	48.95 B	3.3%	33 B	0.68
RedPajama - Books	26.02 B	3%	30 B	1.15
RedPajama - arXiv	28.10 B	1.9%	19 B	0.68
RedPajama - StackExchange	20.54 B	1.4%	14 B	0.68

tokenizer, and the model was pre-trained on 1T tokens sampled according to this mix.

Tokenizer

We used [EleutherAI's GPT-NeoX 20B tokenizer](#). This BPE tokenizer has a number of desirable characteristics, most of which are relevant for tokenizing code:

- Trained on a diverse mix of data that includes code ([The Pile](#))
- Applies consistent space delimitation, unlike the GPT2 tokenizer which tokenizes inconsistently depending on the presence of prefix spaces
- Contains tokens for repeated space characters, which allows superior compression of text with large amounts of repeated space characters.

The tokenizer has a vocabulary size of 50257, but we set the model vocabulary size to 50432. The reasons for this were twofold: First, to make it a multiple of 128 (as in [Shoeybi et al.](#)), which [we found improved MFU by up to four percentage points in initial experiments](#). Second, to leave tokens available that can be used in subsequent [UL2 training](#).

Efficient Data Streaming

We leveraged [MosaicML's StreamingDataset](#) to host our data in a standard cloud object store and efficiently stream it to our compute cluster during training. StreamingDataset provides a number of advantages:

- Obviates the need to download the whole dataset before starting training.
- Allows instant resumption of training from any point in the dataset. A paused run can be resumed without fast-forwarding the dataloader from the start.
- Is fully deterministic. Samples are read in the same order regardless of the number of GPUs, nodes, or CPU workers.
- Allows arbitrary mixing of data sources in: simply enumerate the your data sources and desired proportions of the total training data, and StreamingDataset handles the rest. This made it extremely easy to run preparatory experiments on different data mixes.

Check out the [StreamingDataset blog](#) for more details!

Training Compute

All MPT-7B models were trained on the [MosaicML platform](#) with the following tools:

- **Compute:** A100-40GB and A100-80GB GPUs from Oracle Cloud
- **Orchestration and Fault Tolerance:** [MCLI](#) and [MosaicML platform](#)
- **Data:** OCI Object Storage and [StreamingDataset](#)
- **Training software:** Composer, PyTorch FSDP, and [LLM Foundry](#)

As shown in [Table 3](#), nearly all of the training budget was spent on the base MPT-7B model, which took ~9.5 days to train on 440xA100-40GB GPUs, and cost ~\$200k. The finetuned models took much less compute and were much cheaper – ranging between a few hundred and few thousand dollars each.

Training Costs

Model	Number of Tokens of Data	Train Batch Size (samples)	Train Context Length	System	Time-to-Train with MosaicML	Cost with MosaicML
MPT-7B	1T	1760	2048	440xA100-40GB	9.5 Days	\$200,640
MPT-7B-Instruct	9.6M	48	2048	8xA100-40GB	2.3 Hours	\$37
MPT-7B-Chat	86M	32	2048	8xA100-80GB	8.2 Hours	\$164
			+ 32xA100-40GB		6.7 Hours	\$429
			Total Combined	→	14.9 Hours	\$593
MPT-7B-StoryWriter-65k+	5B	32	65536	32xA100-80GB	2.2 Days	\$4270

 mosaic^{ML} Foundation Series

Table 3 – Training details for each of our MPT-7B models.

'Time to Train' is the total runtime from job start to finish, including checkpointing, periodic evaluation, restarts, etc. 'Cost' is computed with pricing of \$2/A100-40GB/hr and \$2.50/A100-80GB/hr for reserved GPUs on the MosaicML platform.

Each of these training recipes can be fully customized. For example, if you'd like to start from our open source MPT-7B and finetune it on proprietary data with a long context length, you can do that today on the MosaicML platform.

As another example, to train a new model from scratch on a custom domain (e.g. on [biomedical text](#) or [code](#)), simply reserve short-term large blocks of compute with MosaicML's **hero cluster** offering. Just pick the desired model size and token budget, upload your data to an object store like S3, and launch an MCLI job. You will have your very own custom LLM in just days!

Check out [our earlier LLM blog post](#) for guidance on the times and costs to train different LLMs. Find the latest throughput data for specific model configurations [here](#). In line with our previous work, all MPT-7B models were trained with Pytorch FullyShardedDataParallelism (FSDP) and without tensor- or pipeline- parallelism.

Training Stability

As many teams have documented, training LLMs with billions of parameters on hundreds-to-thousands of GPUs is incredibly challenging. Hardware will fail frequently and in creative and unexpected ways. Loss spikes will derail training. Teams must "babysit" the training run 24/7 in case of failures and apply manual interventions when things go wrong. Check out the [OPT logbook](#) for a candid example of the many perils awaiting anyone training an LLM.

At MosaicML, our research and engineering teams have worked tirelessly over the last 6 months to eliminate these issues. As a result, our MPT-7B training logbook ([Figure 5](#)) is very boring! We trained MPT-7B on 1 trillion tokens from start to finish with no human intervention. No loss spikes, no mid-stream learning rate changes, no data skipping, automatic handling of dead GPUs, etc.

```

2023-04-21 12:13 PM - mcloud detects hardware issue [hero-7b-clean-H98p4x]
2023-04-21 12:14 PM - mcloud stops job [hero-7b-clean-H98p4x]
2023-04-21 12:14 PM - mcloud starts job [hero-7b-clean-E4u2bf]
2023-04-22 07:34 AM - mcloud detects hardware issue [hero-7b-clean-E4u2bf]
2023-04-22 07:34 AM - mcloud stops job [hero-7b-clean-E4u2bf]
2023-04-22 07:40 AM - mcloud starts job [hero-7b-clean-pD9RE3]
2023-04-24 04:44 PM - mcloud detects hardware issue [hero-7b-clean-pD9RE3]
2023-04-24 04:44 PM - mcloud stops job [hero-7b-clean-pD9RE3]
2023-04-24 04:49 PM - mcloud starts job [hero-7b-clean-HPpt01]
2023-04-27 01:58 AM - mcloud detects hardware issue [hero-7b-clean-HPpt01]
2023-04-27 01:58 AM - mcloud stops job [hero-7b-clean-HPpt01]
2023-04-27 01:59 AM - mcloud starts job [hero-7b-clean-uK3VDC]
2023-04-29 08:08 AM - user1 completes job [hero-7b-clean]

```

Figure 5 – The (very uneventful) MPT-7B Training Logbook.

MPT-7B was trained on 1T tokens over the course of 9.5 days on 440xA100-40GB. During that time the training job encountered 4 hardware failures, all of which were detected by the MosaicML platform. The run was automatically paused and resumed upon each failure, and no human intervention was required.

Hands-free LLM Training with the mosaic^{ML} Platform

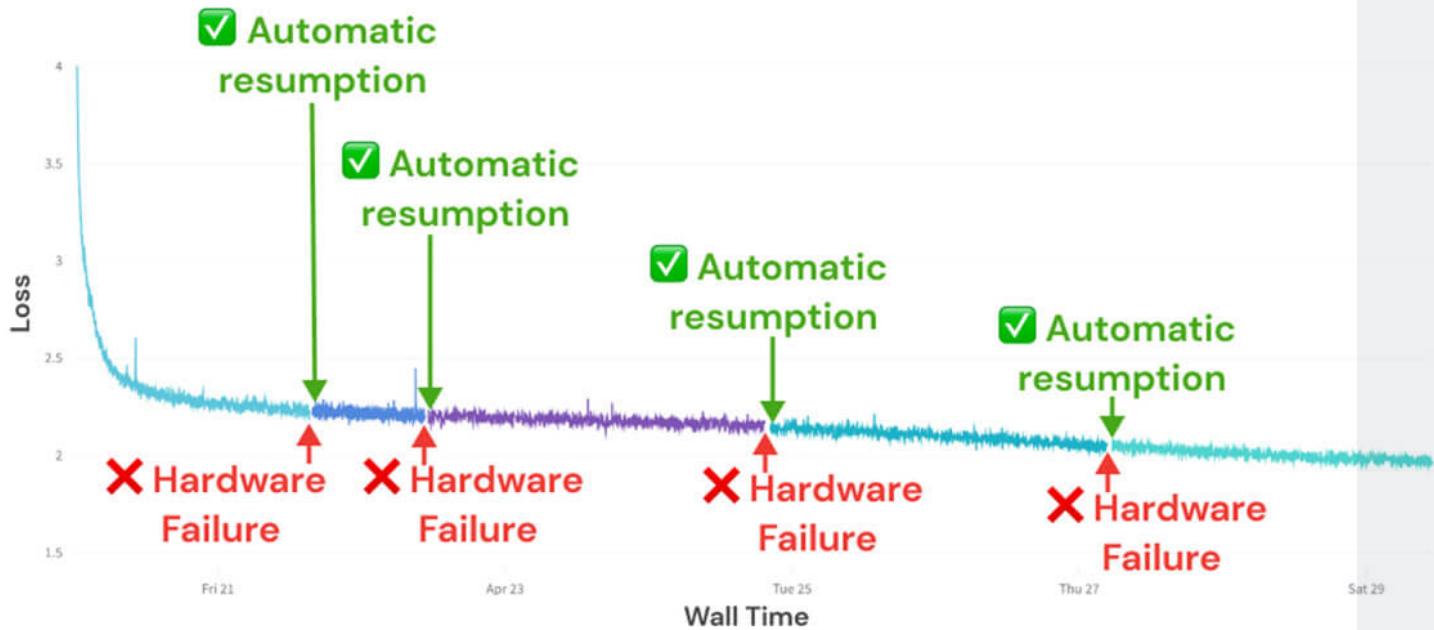


Figure 6 – The loss curve over time, highlighting hardware failures and automatic recoveries.

If hardware failures occur while a job is running, the MosaicML platform automatically detects the failure, pauses the job, cordons any broken nodes, and resumes the job. During the MPT-7B training run, we encountered 4 such failures, and each time the job was automatically resumed

How did we do this? First, we addressed convergence stability with architecture and optimization improvements. Our MPT models use ALiBi rather than positional embeddings, which we found to improve resilience to loss spikes. We also train our MPT models with the Lion optimizer rather than AdamW, which provides stable update magnitudes and cuts optimizer state memory in half.

Second, we used the MosaicML platform's NodeDoctor feature to monitor for and resolve hardware failures and the JobMonitor feature to resume runs after these failures were resolved. These features enabled us to train MPT-7B with no human intervention from start to finish despite 4 hardware failures during the run. See **Figure 6** for a closeup view of what autoresumption looks like on the MosaicML platform.

MPT is designed to be fast, easy, and cheap to deploy for inference. To begin with, all MPT models are subclassed from the [HuggingFace PretrainedModel base class](#), which means that they are fully compatible with the HuggingFace ecosystem. You can upload MPT models to the HuggingFace Hub, generate outputs with standard pipelines like `model.generate(...)`, build HuggingFace Spaces ([see some of ours here!](#)), and more.

What about performance? With MPT's optimized layers (including [FlashAttention](#) and low precision layernorm), the out-of-the-box performance of MPT-7B when using `model.generate(...)` is **1.5x–2x faster** than other 7B models like LLaMa-7B. This makes it easy to build fast and flexible inference pipelines with just HuggingFace and PyTorch.

But what if you **really** need the best performance? In that case, directly port MPT weights to FasterTransformer or ONNX. Check out the [LLM Foundry's inference folder](#) for scripts and instructions.

Finally, for the best hosting experience, deploy your MPT models directly on [MosaicML's Inference service](#). Start with our managed endpoints for models like MPT-7B-Instruct, and/or deploy your own custom model endpoints for optimal cost and data privacy.

What's Next?

This MPT-7B release is the culmination of two years of work at MosaicML building and battle-testing open-source software ([Composer](#), [StreamingDataset](#), [LLM Foundry](#)) and proprietary infrastructure ([MosaicML Training](#) and [Inference](#)) that makes it possible for customers to train LLMs on any compute provider, with any data source, with efficiency, privacy and cost transparency – and to have things go right the first time.

We believe MPT, the MosaicML LLM Foundry, and the MosaicML platform are the best starting point for building custom LLMs for private, commercial, and community use, whether you want to finetune our checkpoints or train your own from scratch. We look forward to seeing how the community builds on these tools and artifacts.

Importantly, today's MPT-7B models are just the beginning! To help our customers address more challenging tasks and continually improve their products, MosaicML will continue to produce foundation models of higher and higher quality. Exciting follow-on models are already training. Expect to hear more about them soon!

Acknowledgements

We are grateful to our friends at AI2 for helping us to curate our pretraining dataset, choose a great tokenizer, and for many other helpful conversations along the way ✌

Appendix

Data

mC4

[Multilingual C4 \(mC4\) 3.1.0](#) is an update of the original [mC4](#) by [Chung et al.](#), which contains sources through August 2022. We selected the English subset, and then applied the following filtering criteria to each document:

- The most common character must be alphabetic.
- $\geq 92\%$ of characters must be alphanumeric.
- If the document is > 500 words, the most common word cannot constitute $> 7.5\%$ of the total word count; If the document is ≤ 500 words, the most common word cannot constitute $> 30\%$ of the total word count.
- The document must be ≥ 200 words and ≤ 50000 words.

mC4 was released as part of the continued effort from [Dodge et al.](#).

C4

[Colossal Cleaned Common Crawl \(C4\)](#) is an English Common Crawl corpus introduced by Raffel et al.. We applied [Abbas et al.'s Semantic Deduplication](#) process to remove the 20% most similar documents within C4, as internal experiments showed that this is a [Pareto improvement](#) for models trained on C4.

RedPajama

We included a number of subsets of the [RedPajama dataset](#), which is [Together](#)'s attempt to replicate LLaMA's training data. Specifically, we used the CommonCrawl, arXiv, Wikipedia, Books, and StackExchange subsets.

The Stack

We wanted our model to be capable of code generation, so we turned to [The Stack](#), a 6.4TB corpus of code data. We used [The Stack Dedup](#), a variant of the stack that has been approximately deduplicated (via MinHashLSH) to 2.9TB. We selected a subset of 18 of The Stack's 358 programming languages in order to reduce dataset size and increase relevance:

- C
- C-Sharp
- C++
- Common Lisp
- F-Sharp
- Fortran
- Go
- Haskell
- Java
- Ocaml
- Perl
- Python
- Ruby
- Rust
- Scala
- Scheme
- Shell
- Tex

We chose to have code constitute 10% of the pretraining tokens, as internal experiments showed that we could train on up to 20% code (and 80% natural language) with no negative impact on natural language evaluation.

We also extracted the Markdown component of The Stack Dedup and treated this as an independent pretraining data subset (i.e. not counted towards the 10% code tokens). Our motivation for this is that markup language documents are predominantly natural language, and as such should count towards our natural language token budget.

Semantic Scholar ORC

The Semantic Scholar Open Research Corpus (S2ORC) is a corpus of English-language academic papers, which we consider to be a high-quality data source. The following quality filtering criteria were applied:

- The paper is in English (as assessed using [cld3](#)).
- The paper has at least 500 words and 5 paragraphs.
- The paper was published after 1970 and before 2022-12-01.
- The most frequent word in the paper consists of alpha characters only, and it appears in less than 7.5% of the document.

This yielded 9.9M papers. Instructions to obtain the latest dataset version are available [here](#), and the original publication is [here](#). The filtered version of the dataset was kindly provided to us by AI2.

Evaluation Tasks

Lambada: 5153 samples of text curated from the books corpus. Consists of a several hundred word paragraph in which the model is expected to predict the next word.

PIQA: 1838 samples of physical intuitive binary multiple choice questions, e.g. "Question: How can I easily carry clothes on hangers when I move?", "Answer: "Take a couple of empty heavy duty clothes hangers, then hook several hangers of clothes on Those hangers and carry them all at once."

COPA: 100 sentences of the form XYZ therefore/because TUV. Framed as binary multiple choice questions where the model has a choice of two possible ways to follow the therefore/because. e.g. {"query": "The woman was in a bad mood, therefore", "gold": 1, "choices": ["she engaged in small talk with her friend.", "she told her friend to leave her alone."]}

BoolQ: 3270 yes/no questions based on some passage which contains relevant information. Question topics range from pop culture to science, law, history, etc. e.g. {"query": "Passage: Kermit the Frog is a Muppet character and Jim Henson's most well-known creation. Introduced in 1955, Kermit serves as the straight man protagonist of numerous Muppet productions, most notably Sesame Street and The Muppet Show, as well as in other television series, films, specials, and public service announcements through the years. Henson originally performed Kermit until his death in 1990; Steve Whitmire performed Kermit from that time up until his dismissal from the role in 2016. Kermit is currently performed by Matt Vogel. He was also voiced by Frank Welker in Muppet Babies and occasionally in other animation projects, and is voiced by Matt Danner in the 2018 reboot of Muppet Babies.\nQuestion: has kermit the frog been on sesame street?\n", "choices": ["no", "yes"], "gold": 1}

Arc-Challenge: 1172 challenging four-choice multiple choice questions about science

Arc-Easy: 2376 easy four choice multiple choice science questions

HellaSwag: 10042 four choice multiple choice questions in which a real life scenario is presented and the model must choose the most likely conclusion to the scenario.

Jeopardy: 2117 Jeopardy questions from five categories: science, world history, us history, word origins, and literature. The model must provide the exact correct answer

MMLU: 14,042 multiple choice questions from 57 diverse academic categories

TriviaQA: 11313 free response pop culture trivia questions

Winograd: 273 schema questions where the model must resolve which referent of a pronoun is most likely.

Winogrande: 1,267 schema questions where the model must resolve which ambiguous sentence is more logically likely (both versions of the sentence are syntactically valid)

MPT Hugging Face Spaces Privacy Policy

Please see our [MPT Hugging Face Spaces Privacy Policy](#).

[Try Databricks for free](#)[Get Started](#)

Related posts



MPT-30B: Raising the bar for open-source foundation models

June 22, 2023 by [The MosaicML NLP Team](#) in [Generative AI](#)

Introducing MPT-30B, a new, more powerful member of our Foundation Series of open-source models, trained with an 8k context length on NVIDIA H100...



Announcing MPT-7B-8K: 8K Context Length for Document Understanding

July 18, 2023 by [Sam Havens](#) and [Erica Ji Yuen](#) in [Generative AI](#)

Today, we are releasing MPT-7B-8K, a 7B parameter open-source LLM with 8k context length trained with the MosaicML platform. MPT-7B-8K was pretrained starting...



MosaicBERT: Pretraining BERT from Scratch for \$20

March 9, 2023 by [Jacob Portes](#), [Alex Trott](#), [Daniel King](#), [Sam Havens](#) and [Erica Ji Yuen](#) in [Generative AI](#)

With the MosaicBERT architecture + training recipe, you can now pretrain a competitive BERT-Base model from scratch on the MosaicML platform for \$20...

[See all Generative AI posts](#)



[Why Databricks](#) ▾

[Product](#) ▾

[Solutions](#) ▾

[Resources](#) ▾

[About](#) ▾

Databricks Inc.
160 Spear Street, 13th Floor
San Francisco, CA 94105
1-866-330-0121

See Careers
at Databricks

© Databricks 2024. All rights reserved. Apache, Apache Spark, Spark and the Spark logo are trademarks of the Apache Software Foundation.

[Privacy Notice](#) | [Terms of Use](#) | [Your Privacy Choices](#) | [Your California Privacy Rights](#)
| [Health and Safety Statement of Intent](#)  