Beens-MiniMax: How not to Build an LLM

Abinesh Mathivanan

Beens-Minimax is a 103-million-parameter SLM (Small Language Model) with a hybrid attention system (Softmax and Lightning), MoE (Mixture of Experts) layers, pretrained with Wikitext-103 for 15 GPU hours (+6 hours for Instruct-SFT) in a Kaggle restricted environment (2x T4 GPUs). I inspired the architecture from the original MiniMax-01 model (https://github.com/MiniMax-AI/MiniMax-01) and attempted to recreate it as a learning project. In this report, I'll explain the issues I had to resolve while training this and how not to build an SLM like me, as this doesn't even generate any sensible text, which I could even benchmark against the outdated datasets.

1. Introduction

I don't have much introduction to explain. Just get into the next parts, you'll eventually understand.

Github : Beens-MiniMax
Kaggle (Base Model) : Beens-MiniMax Base Model
Kaggle (notebook implementation) : Beens-MiniMax Notebook

2. Model Architecture

I started with understanding the architecture of the MiniMax block as represented below 1, and with my current understanding, the model works in the way,

I made Beens-Minimax as a decoder-only transformer model with approximately 103.2 million parameters. The architecture is a stack of $L = 8$ identical decoder blocks. Each decoder block, indexed by l, processes an input sequence of token embeddings $X^{(l-1)} \in \mathbb{R}^{N \times d}$, where N is the sequence length and d is the model's hidden dimension (512). The processing flow follows a Post-LayerNorm structure with DeepNorm scaling.

1. **Pre-Attention Normalization:** The input $X^{(l-1)}$ is first normalized.
$$\hat{X}^{(l-1)} = \text{RMSNorm}(X^{(l-1)}) \tag{1}$$
2. **Multi-Head Attention:** The normalized input is processed by the attention sub-layer.
$$A^{(l)} = \text{Attention}(\hat{X}^{(l-1)}) \tag{2}$$
3. **First Residual Connection:** A residual connection with DeepNorm scaling is applied. α is the DeepNorm scaling factor.
$$H^{(l)} = X^{(l-1)} + \alpha \cdot A^{(l)} \tag{3}$$
4. **Pre-FFN Normalization:** The output of the attention sub-layer is normalized again.
$$\hat{H}^{(l)} = \text{RMSNorm}(H^{(l)}) \tag{4}$$
5. **Mixture of Experts (MoE) FFN:** The normalized tensor is processed by the MoE feed-forward network.
$$F^{(l)}, R^{(l)} = \text{MoE}(\hat{H}^{(l)}) \tag{5}$$
where $R^{(l)}$ are the router logits used for the auxiliary loss.

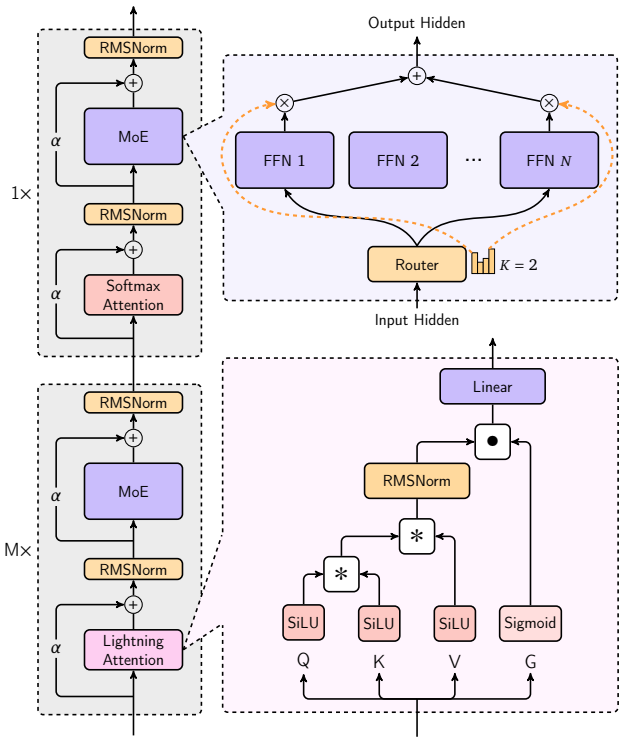

Figure 1 | **The architecture of MiniMax-01 (image taken from MiniMax-01.)**

6. **Second Residual Connection:** The final residual connection is applied to produce the output of the block.

$$X^{(l)} = H^{(l)} + \alpha \cdot F^{(l)} \tag{6}$$

This output $X^{(l)}$ then serves as the input to the next block, $l+1$.

2.1. A Review on Components

2.1.1. RMS Normalization (RMSNorm)

For a given input vector $\mathbf{x} \in \mathbb{R}^d$, RMSNorm is defined as:

$$\text{RMSNorm}(\mathbf{x}) = \frac{\mathbf{x}}{\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2 + \epsilon}} \cdot \mathbf{g} \tag{7}$$

where $\mathbf{g} \in \mathbb{R}^d$ is a learnable gain parameter and ϵ is a small constant for numerical stability. Unlike Layer Normalization, RMSNorm does not perform mean-centering, which simplifies the computation while maintaining effective stabilization.

2.1.2. Rotary Position Embedding (RoPE)

RoPE encodes absolute positional information by rotating a fraction of the query and key vectors in the attention mechanism. For a position m and a feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, RoPE is applied to pairs of features (x_{2i-1}, x_{2i}). The transformation is equivalent to multiplication by a rotation matrix in the complex plane:

$$\begin{pmatrix} x'_{2i-1} \\ x'_{2i} \end{pmatrix} = \begin{pmatrix} \cos(m\theta_i) & -\sin(m\theta_i) \\ \sin(m\theta_i) & \cos(m\theta_i) \end{pmatrix} \begin{pmatrix} x_{2i-1} \\ x_{2i} \end{pmatrix} \tag{8}$$

where the frequency $\theta_i = b^{-2i/d_{rot}}$ for a chosen base b (e.g., 10000) and rotational dimension d_{rot}. This ensures that the dot product between a query at position m and a key at position n depends only on their relative position $m - n$. In our model, this is applied to 50% of the head dimension.

2.1.3. Softmax Attention

Used periodically (layers 4 and 8), this is the standard scaled dot-product attention. For a set of queries Q, keys K, and values V, the output is:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{9}$$

where d_k is the dimension of the key vectors. Our implementation uses Grouped-Query Attention (GQA), reducing memory bandwidth for the KV cache.

2.1.4. Lightning Attention

I have never read this before, so I didn't make this implementation in a properly way. The remaining layers employ a gated linear attention mechanism. The core idea is to achieve linear complexity to sequence length N. The simplified, non-causal form can be expressed as:

$$\text{LinearAttn}(Q, K, V) = \phi(Q)(\phi(K)^T V) \tag{10}$$

where ϕ is a kernel function (e.g., SiLU). Our implementation adds a learnable gate G to modulate the output:

$$\text{Output} = G \odot \text{LinearAttn}(Q, K, V) \tag{11}$$

where $\odot$ denotes element-wise multiplication. I didn't write optimized Triton kernels to support this.

3. Training and Fine-Tuning

The training phase was conducted over a period of **5 days (4 days + 1 day)** by storing the model using epoch and step checkpoints in my drive.

3.1. Phase 1: Pre-training

I pre-trained the base model with the following config,

- **Epochs:** 3 (15 GPU Hours)
- **Steps:** 28k each
- **Corpus:** 1.8M rows of WikiText-103.
- **Optimizer & Scheduler:** AdamW optimizer was used. To overcome the loss plateau observed after approximately 30,000 steps, a Cyclical Learning Rate scheduler was implemented, which varies the LR between 1×10^{-5} and 3×10^{-4}, forced the model out of local minima and encouraged continued, albeit slow, convergence.
- **Loss Functions:** Cross-Entropy Loss and MoE Load balancing Loss
- **Hardware:** Training was distributed across two NVIDIA T4 GPUs using 'torch.nn.DataParallel'.

3.2. Phase 2: Supervised Fine-Tuning (SFT)

The pre-trained model was then adapted into a conversational agent.

- **Epochs:** 2 (6 GPU Hours)
- **Steps:** 28k each
- **Dataset:** A 200k dataset of HuggingFaceH4/ultrachat_200k.
- **Technique:** LoRA was used to fine-tune the model. The base model's weights were frozen, and only the injected low-rank adapter matrices were trained. This reduced the number of trainable parameters to less than 0.5% of the total, which is 500k parameters.
- **Learning Rate:** 2e-4
- **Loss Function:** Cross-Entropy Loss
- **Prompt Engineering:** Each sample was formatted into a `### HUMAN:` / `### ASSISTANT:` template to teach the model the turn-taking structure of dialogue.

3.3. Model Configuration

Parameter	Value
Core Architecture	
Vocabulary Size ('vocab_size')	50,257 (from 'gpt2' tokenizer)
Hidden Size ('hidden_size')	512
Number of Layers ('num_layers')	8
Intermediate FFN Size ('ffn_hidden_dim')	2048
Attention Mechanism	
Attention Heads ('num_attention_heads')	8
Head Dimension ('head_dim')	64
GQA Group Size ('gqa_group_size')	2
Key/Value Heads ('num_key_value_heads')	4
Softmax Attention Period	Every 4th layer
MoE Layer	
Number of Experts ('num_experts')	4
Experts per Token ('num_experts_per_tok')	2 (top-2 routing)
Architectural Details	
RoPE Base Frequency ('rope_base')	10,000
RoPE Applied Fraction ('rope_dim_fraction')	0.5
DeepNorm Alpha ('deepnorm_alpha')	≈ 2.0 $((2 \times 8)^{0.25})$
RMSNorm Epsilon ('rms_norm_eps')	1×10^{-6}
MoE Aux Loss Coefficient ('router_aux_loss_coef')	0.01

Table 1 | **Beens-MiniMax Model Configuration**

4. Observations and Analysis

4.1. Pre-training Loss Plateau at 3.3

During the pre-training phase, a flat loss plateau was observed. After an initial period of rapid convergence, the training loss ceased to decrease significantly after approximately 30,000 steps, fluctuating between 3.3 and 3.9 for the remainder of the three epochs 2. I derive that the observed phenomenon is due to the following factors,

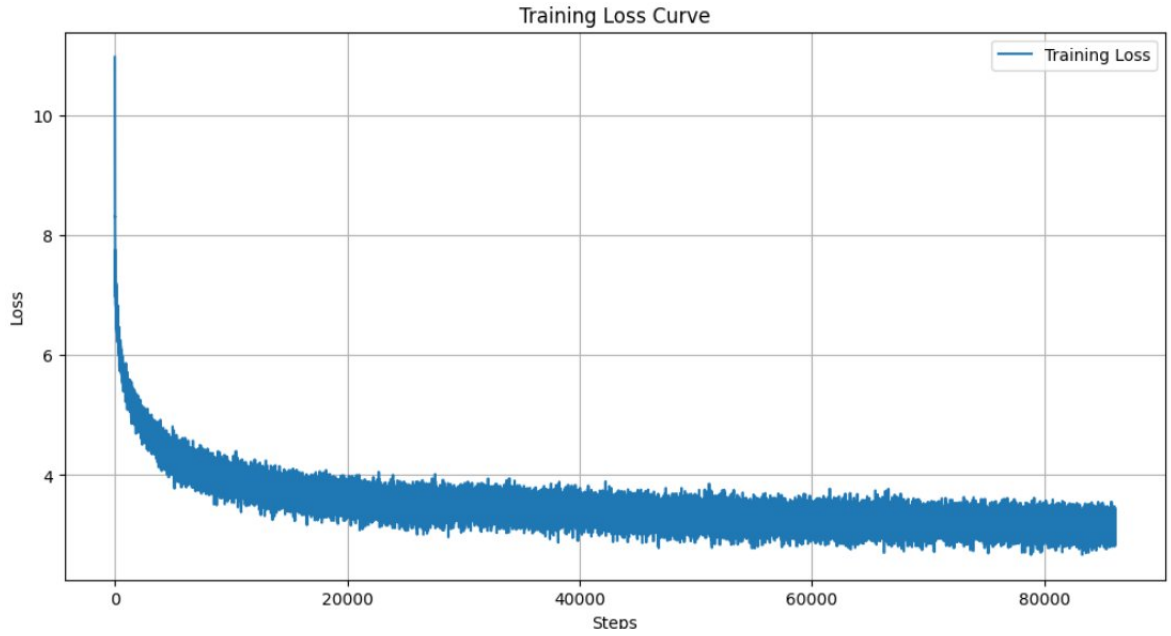

Figure 2 | **Training loss curve for Beens-MiniMax (Base Model)**

1. **Model Capacity Limit:** The 103M parameter architecture, while capable, possesses a finite capacity for knowledge representation. I hypothesize that after 30k steps, the model had successfully learned the most statistically significant patterns in the WikiText-103 dataset. The loss value of 3.3 likely represents the "informational floor" or irreducible error for a model of this specific size on this specific dataset. We will not be able to push the loss below after this with the current number of parameters.
2. **Dataset Homogeneity:** WikiText-103 is a high-quality corpus, but homogeneous compared to massive web-scale datasets. After three epochs, the model has seen every token sequence multiple times. The model may have extracted nearly all the unique information it can from the dataset, leading to diminishing returns on further training. The plateau is not due to "improper" data, but rather the exhaustion of "learnable" information in the dataset.

4.2. Ineffectiveness of Cyclical LR Below a 2.9 Loss

While implementing a Cyclical Learning Rate (CLR) did help the model break out of minor plateaus and continue its descent, we observed that it was unable to push the loss below a hard floor of approximately 2.9, even at the lowest points of the learning rate cycle.

I think the cyclical increases in learning rate are effective at "kicking" the optimizer out of poor local minima and navigating the loss landscape. However, if the landscape itself has a "global" valley for this model-data combination that is no deeper than 2.9, no amount of optimizer momentum can overcome that fundamental limit. The CLR ensures we are exploring the valley thoroughly, but it cannot make the valley itself deeper. With this, we could conclude that the loss of 2.9 is most probably the best case performance for this limitation of parameters.

4.3. LoRA SFT Loss Fluctuation

During the SFT phase with LoRA, the training loss was highly volatile, fluctuating heavily between 3.2 and 4.5 throughout the two training epochs 3.

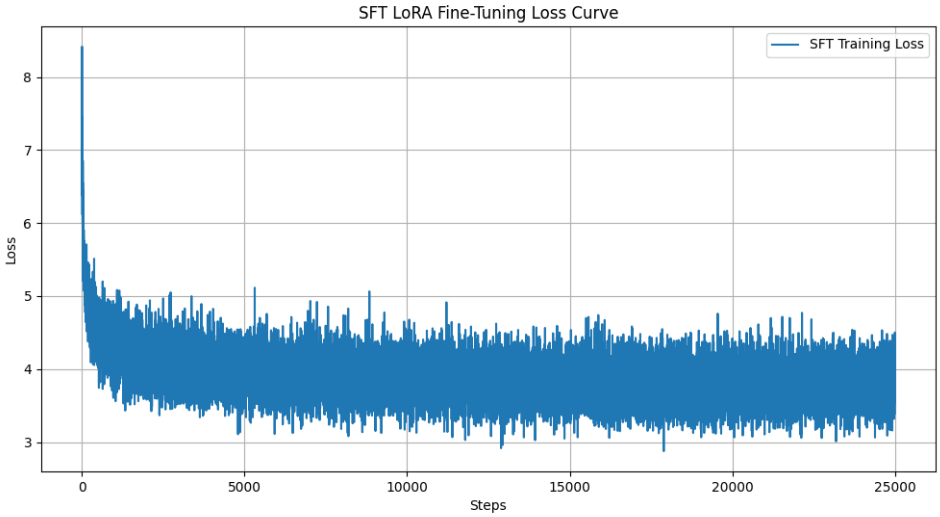

Figure 3 | **Training loss curve for Beens-MiniMax (Instruct-SFT)**

I observe that the behavior might be cause of the following factors,

- **Task Mismatch:** The base model was pre-trained as a text completer following a very similarly structured raw corpus. The SFT task requires it to be an instruction-following chatbot. Each batch from ‘Ultrachat_200k‘ presents a completely different "task" (e.g., one batch is about creative writing, the next about factual Q&A). This leads to a massive behavioural shift.
- **Small Adapters, High Learning Rate:** LoRA adapters have very few parameters. A relatively high learning rate (2e-4) causes large gradient updates to these few parameters. When a batch contains a task that is very different from the previous one, the adapters can be "pulled" aggressively in a new direction, causing a sharp spike in the loss.
- **Limited Training:** As the adapter was only trained for 2 epochs, it’s possible that it had learned the structure of the responses but not the actual context of the responses. We could try increasing the adapter parameters along with task-specific loss functions and efficient optimizers (like Muon).

4.4. Machine Unlearning: Degradation in Long-Context Generation

This was my most interesting finding. After the second pre-train epoch, the model became highly proficient at short, conversational Question-Answering. However, its ability to generate long, coherent, creative text (a skill it possessed after pre-training) degraded significantly. By the end of the third pre-training epoch, the model became unstable, often generating repetitive loops or ‘<unk>‘ tokens.

I observe this as a case of **catastrophic forgetting** and **overfitting** due to the intensive CLR rates.

- **Epoch 2: Stable Base Model:** For the first two epochs, the model behaved as expected. The training loss decreased steadily before settling into a stable plateau around a value of 3.3. At the end of the second epoch, the model was a competent and stable text completer. When prompted with the beginning of a sentence, it could generate long stable text.
- **Epoch 3: Forgetting and Memorization:** After two full passes over the WikiText-103 dataset, the model had likely learned most of the generalizable linguistic patterns. During the third epoch, instead of learning new patterns, it began to memorize the specific idiosyncrasies and

vocabulary of the training set. Its worldview narrowed from "how English works" to "how WikiText-103 is written."

- **The ‘<unk>‘ Token and Instability:** The increased generation of unknown tokens and repetitive loops is a symptom of this overfitting. The model’s probability distributions become "spiky," favoring the limited vocabulary and patterns seen in the tokenizer. When forced to generate text outside of this narrow domain, its internal states become unstable, leading to decoding failure. The high learning rate in the later stages of training exacerbated this, causing the adapter weights to become too specialized and brittle.

5. Thanks

I took this report template from the MiniMax arxiv publication (https://arxiv.org/abs/2501.08313).

A. Beens-MiniMax Base Model Demonstration

We define the following testing config,
Temperature = 0.8
Max Length = 100
top_k = 50

Long Context Generation (Language)

User Request (Text completion - 100 tokens)

Once upon a time, there was a kingdom

Beens-Minimax

Once upon a time, there was a kingdom where its inhabitants would also have to take place . Some people would have been forced to settle in a place called " Castle Rock " . In addition , the remains of the city were found to be buried in a large , permanent , and permanent structure . There is a second possible tomb , built between 1040 and 1545 by John de <unk>

Long Context Generation (Science)

User Request (Text completion - 100 tokens)

The equation $E = mc^2$, derived by Albert Einstein, means that

Beens-Minimax

The equation $E = mc^2$, derived by Albert Einstein, means that the planet is not only a single planet with an eccent ric orbit , but a system with the same name , the Hubble telescopes . The term " O " in the table , for example , is used only for the planets to be discovered , and thus not the first planets , as these planets have a differen t composition and structure ,

Long Context Generation (History)

User Request (Text completion - 100 tokens)

The history of the Roman Empire is a fascinating subject, particularly the transition from the Republic to the Principate, which began

Beens-Minimax

The history of the Roman Empire is a fascinating subject, particularly the transition from the Republic to the Principate, which began in the 19th century . The first major source of the Empire was the rise of the British Empire in the early 18th century by the British naval forces . The French Navy had the opportunity to build a naval fleet , the most powerful naval theatre in the world , as it was also the first capital ship to be built . The French Navy , however , had only a few

As you can see, it's stupid.

B. Beens-MiniMax Instruct-SFT Demonstration

I trained the Base model with Huggingface's UltraChat_200k (Dataset) for 6 GPU Hours and tried to make it an Instruct model.

We define the following testing config,
Temperature = 0.8
Max Length = 150
top_k = 50

Question & Answering

User Request

What is the capital of France?

Beens-Minimax

The capital of Belgium is the capital of Belgium.

Question & Answering

User Request

Hello. What can you do?

Beens-Minimax

1. Set a new level of interest and experience in the same way?
2. The second level of interest: For the first level of interest and experience in the second level, the second level will be filled with a new level of interest in the second level.
3. The position: The second level will be filled with an upper-class perspective, but the screen will be filled with a new level of interest to the next level.
4. The position: The second level will be filled with a new level of interest.
5. The center will be filled with a new level of interest. The position

Question & Answering

User Request

Write a short, three-line poem about a cat.

Beens-Minimax

I don't have a cat.
The cat's world is forever and we are a great place to explore the ways in which we live, and it's a great place to explore the world.
The cat's world is a place to explore the world, and we can provide our own insight into our lives.
The cat's world is a unique place to explore the world. The stars are the same size, but we have a cat's ability to focus on the world, who are the most vulnerable to our lives.