Small-Scale Foundation Model Training — Report

1. Model Architecture and Parameters

MiniGPT is a decoder-only transformer for next-token prediction (language modeling). It is implemented in `model.py` and configured via `config.py`.

Architecture

- **Token embedding**: `nn.Embedding(vocab_size, embed_dim)` — no shared padding index in the current implementation.
- **Positional encoding**: Learned positional embeddings added to token embeddings (`PositionalEmbedding`: `nn.Embedding(max_len, embed_dim)`).
- **Transformer blocks** (repeated `n_layers` times):
  - **Pre-norm**: LayerNorm → Multi-Head Self-Attention (residual) → LayerNorm → FFN (residual).
  - **Multi-Head Attention**: Single linear projects to Q, K, V; `n_heads` heads; scaled dot-product attention; optional causal/padding mask; output projection.
  - **FFN**: Two linear layers with hidden size `embed_dim * ffn_mult` and GELU activation.
- **Output**: Final LayerNorm and a linear LM head (`embed_dim → vocab_size`, no bias) producing logits over the vocabulary.

Baseline Parameters

Parameter	Value
`vocab_size`	50,257
`seq_len`	64
`embed_dim`	128
`n_heads`	4
`n_layers`	2
`ffn_mult`	4 (FFN hidden = 512)
`dropout`	0.1 (in attention; not applied in FFN in current code)

Head dimension is `embed_dim // n_heads` (e.g. 32 for baseline). Total parameter count scales with `embed_dim`, `n_layers`, and `vocab_size`.

2. Dataset Details

- **Source**: Data is loaded from assignment01:
  - **Primary**: `tokenized_data.pt` — list of token id tensors (variable-length).
  - **Fallback**: `sample_dataset.pt` — batched data with `input_ids`.
- **Vocabulary**: GPT-2 style; `VOCAB_SIZE = 50257`, `PAD_TOKEN_ID = 50256`.
- **Sequence length**: Fixed at `SEQ_LEN = 64`. Valid sequence length range is `SEQ_MIN = 32` to `SEQ_MAX = 128`. Sequences are truncated or split into chunks of `SEQ_LEN`; shorter sequences are padded with `PAD_TOKEN_ID` to length 64.
- **Batching**: `SeqDataset` yields sequences of length `SEQ_LEN`; `DataLoader` uses configurable `batch_size` (e.g. 32 for baseline) with shuffle and 4 workers.

---

3. Training Setup and Hyperparameter Experiments

Training Setup

- **Objective**: Next-token prediction. Logits from positions `0 .. L−2` are compared to labels at positions `1 .. L−1`; loss is computed with `CrossEntropyLoss(ignore_index=PAD_TOKEN_ID)` so padding positions do not contribute to the loss.
- **Optimizer**: AdamW with learning rate set per experiment.
- **Epochs**: 1 (single full pass over the dataset).
- **Metrics**: Training loss and perplexity are recorded **per step** (per batch): `perplexity = exp(loss)`.
- **Checkpoints**: Model state dict is saved under `outcome/` as `mini_gpt_<experiment_name>.pt`.
- **Device**: Each experiment can be assigned a GPU (e.g. `cuda:1` – `cuda:5`) via `run_experiments.py --devices 1,2,3,4,5`; experiments run sequentially, each on its designated device.

Hyperparameter Experiments

Experiment	Override(s)	Description
baseline	—	Default: lr=5e-4, batch_size=32, n_layers=2, embed_dim=128, n_heads=4, max_epochs=1.
lr_1e-3	lr=1e-3	Higher learning rate.
batch_size_16	batch_size=16	Smaller batch size

batch_size_16	batch_size=16	Smaller batch size.
n_layers_1	n_layers=1	Single transformer layer.
embed_dim_64	embed_dim=64	Smaller embedding and hidden size (n_heads=4 ⇒ head_dim=16).

Plots and tables are written to `outcome/`: per-experiment curves (`loss_curve_<name>.png`, `perplexity_curve_<name>.png`), comparison curves (`loss_curves.png`, `perplexity_curves.png`), and summary tables (`results_table.txt`, `results_table.csv`).

---

4. Observations and Challenges

- **Single epoch**: With only one epoch, curves show how quickly each setup descends in one pass; comparison across experiments is limited to this short horizon.
- **Multi-GPU usage**: Experiments are run one after another, each on a different GPU when `--devices` is set; true parallel multi-GPU training (e.g. one process per experiment) would require separate processes or launcher scripts.
- **Reporting**: The report summarizes the model, data pipeline, training configuration, and the listed hyperparameter experiments; numerical results and plots in `outcome/` should be consulted for concrete numbers and curves.