Phoenix-SLM (Small Language Model)

"PSM"

Pattern-Synthesis Model

Aditya Mishra

github.com/aam-007/psm

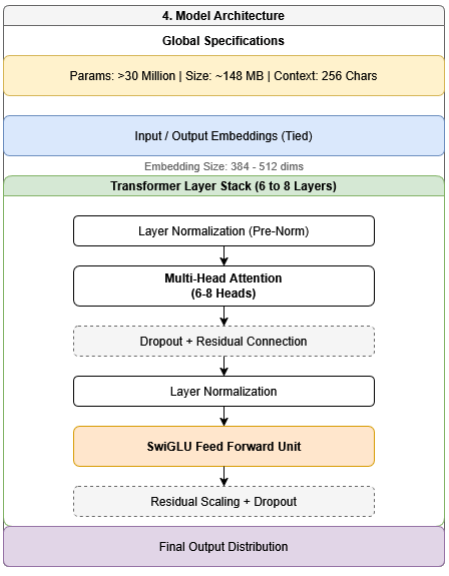
Abstract. This paper presents the Pattern Synthesis Model "PSM", a compact transformer based Small Language Model designed to explore how linguistic structure, reasoning patterns, and stylistic identity can emerge from an extremely small dataset. "PSM" is a 148 MB model trained entirely on approximately 60 KB of personal essays written by a single author. The goal of this work is not scale, but fidelity. The project examines whether a small model can compress a consistent body of thought into a functional system that generates coherent language while retaining the author's characteristic voice and introspective tone. "PSM" demonstrates that even with limited data, a carefully designed architecture and a consistent dataset can produce a stable and expressive model.

1. Introduction

Language models are typically associated with enormous datasets and large parameter counts. This trend has produced highly capable systems, but it has also created a distance between a model and any specific human author. Large corpora dilute personal style. They also demand extensive compute resources that are inaccessible for most individuals.

"PSM" aims to explore the opposite direction. Instead of breadth, the model focuses on depth within a single writer's worldview. The training dataset consists exclusively of personal essays, reflections, and short philosophical notes. This creates an unusually consistent dataset. Because of this consistency, the model is able to learn not only surface level phrasing but also the patterns of reasoning and rhythm of thought that run through the original texts.

The central idea behind "PSM" is simple. If a dataset contains a clear cognitive fingerprint, then a model does not need scale to learn it. It only needs to be designed with enough capacity and trained with enough stability to extract patterns that matter.



2. Design Goals

“PSM” was built to answer three questions:

1. How much reasoning depth can a small model achieve if the dataset is internally consistent.
2. Can a model learn a recognizable voice from a very small corpus.
3. What architectural features matter most when the primary limitation is the dataset, not the model size.

The intention is not to create a general-purpose assistant. Instead, “PSM” serves as an experiment in personal cognitive modelling. It is a system shaped by one author’s logic, writing style, and structure of introspection.

3. Dataset

The model was trained on approximately 60 KB of text. The dataset includes:

- personal essays
- philosophical reflections
- analytical paragraphs
- short introspective notes

The writing style across the dataset is highly uniform. This uniformity is essential for a project of this scale. A model cannot infer stable reasoning from scattered or inconsistent text. A compact dataset can still carry strong internal patterns if the author writes with clarity and structure.

The dataset is processed at the character level. This choice avoids the need for tokenization heuristics and allows the model to learn grammar, spacing, punctuation, and rhythm directly from raw text.

4. Model Architecture

“PSM” uses a transformer architecture designed for stability, clarity, and efficient pattern extraction. Key architectural components include:

- 6 to 8 transformer blocks
- 6 to 8 attention heads
- embedding dimensions between 384 and 512
- context window of 256 characters
- SwiGLU feed forward layers
- LayerNorm on all block inputs

- Residual scaling parameters for smooth training
- Tied token embedding and output projection weights
- Top k sampling and temperature control for generation

The total parameter count is above 30 million, resulting in a 148 MB model when saved. This size allows the network to store and compress stylistic features without relying on external corpora.

5. How It Works

“PSM” operates through a three-stage process: encoding, pattern transformation, and autoregressive generation.

5.1 Encoding

Each character in the input is mapped to an integer through a simple character level vocabulary. These integers are then converted into dense vectors through the token embedding table. Positional embeddings are added to provide information about the location of each character in the sequence. The result is a sequence of vectors that represent both content and position.

5.2 Pattern Transformation

The embedded sequence is passed through multiple transformer blocks. Each block applies two key operations:

1. **Self-Attention**
   The model compares every position in the sequence with every other position. This allows it to learn long range relationships, structural patterns, and stylistic tendencies. For example, it learns how an idea introduced early in a paragraph might influence the structure of sentences that follow.

2. **SwiGLU Feed Forward Layers**
   These layers project the representations into higher dimensional spaces, apply a gated nonlinear transformation, and compress them back. This helps the model capture subtle relationships between abstract concepts that appear across the author’s writing style.

Layer normalization and residual connections maintain stability during training. Residual scaling parameters further reduce the risk of training collapse in small data settings.

5.3 Autoregressive Generation

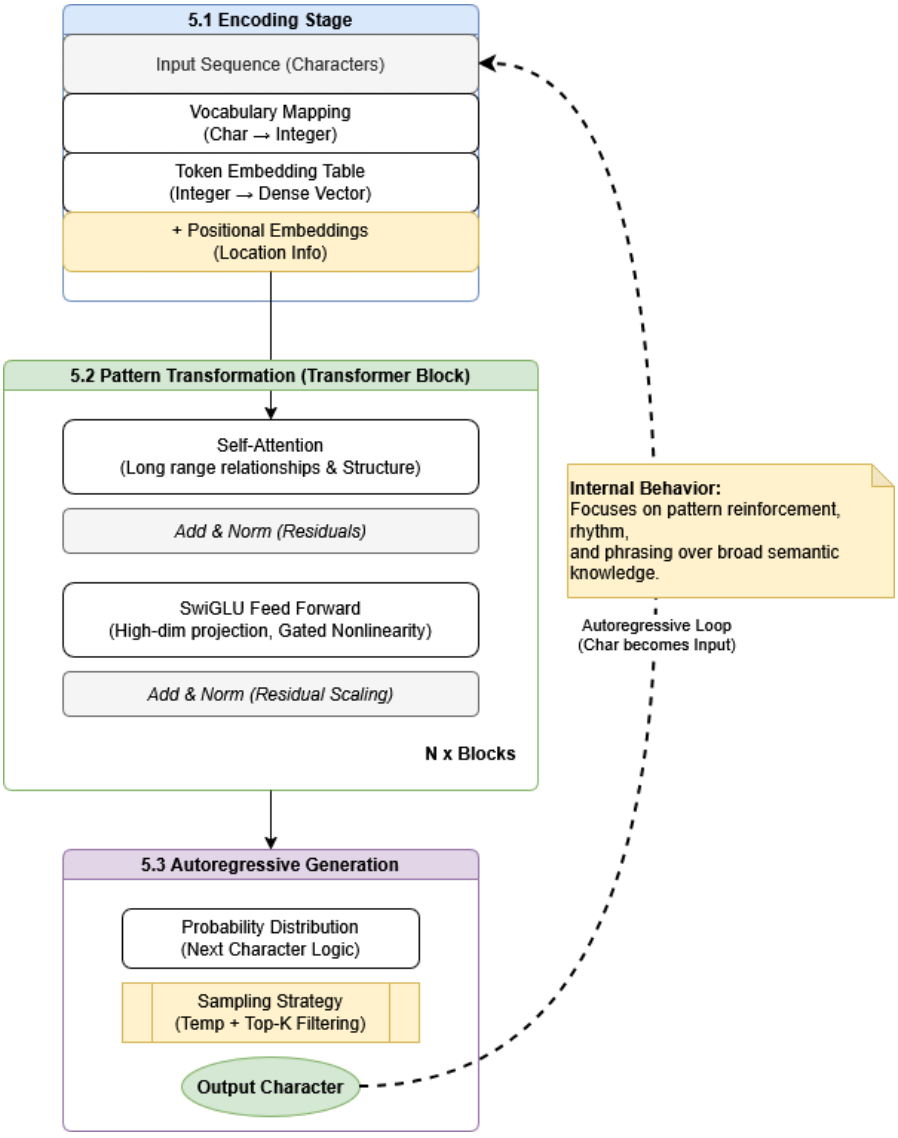
After the final transformer block, the model produces a probability distribution over possible next character. During generation, the model samples from this distribution using temperature and top k filtering. Temperature controls randomness and top k prevent the model from choosing extremely unlikely characters.

This process repeats one character at a time. The model builds text sequentially, treating each generated character as part of the new input. Because the training data is small but consistent, the model tends to reproduce a clear and unified voice that mirrors the author.

5.4 Internal Behaviour

Since the dataset is narrow, the model focuses heavily on pattern reinforcement rather than broad semantic knowledge. It learns the author’s rhythm of argumentation, phrasing habits, and

preferred transitions. As a result, the system generates text that feels introspective and structured. It does not rely on external facts or internet style patterns. Its behavior emerges from the internal coherence of the dataset.

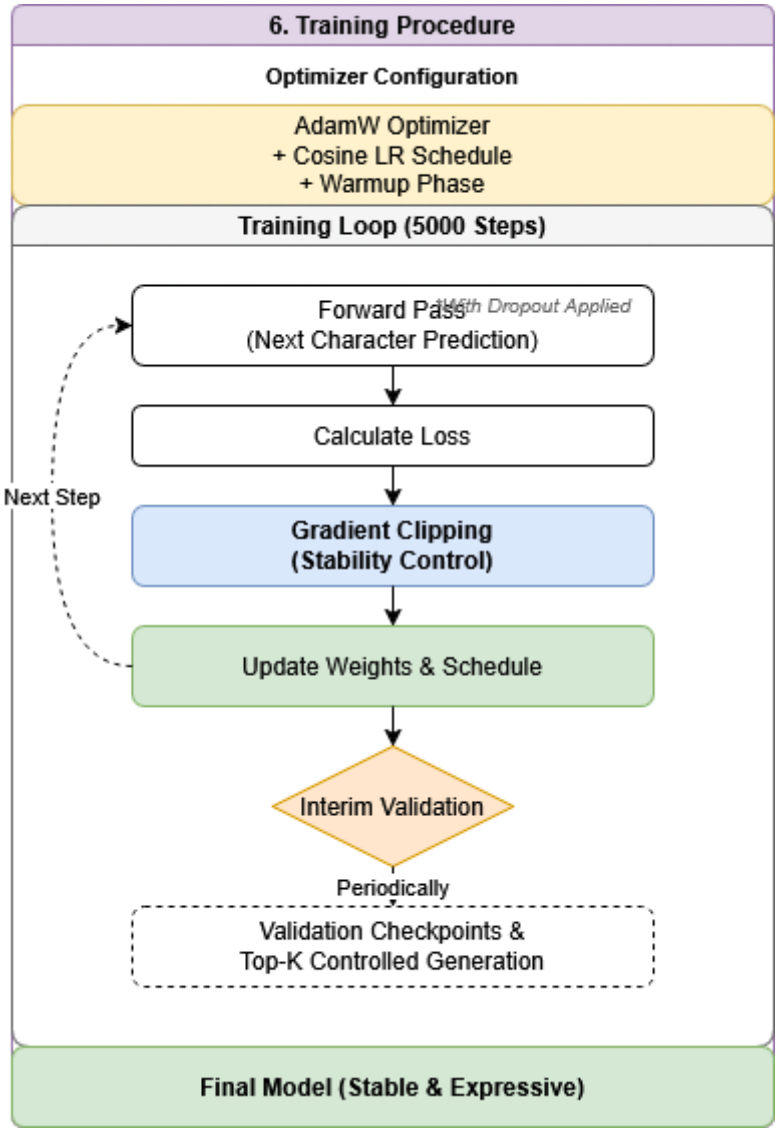


6. Training Procedure

The model is trained with next character prediction. The training loop uses:

- AdamW optimizer
- cosine learning rate schedule
- learning rate warmup
- gradient clipping
- dropout
- validation checkpoints
- top k sampling for controlled interim generation

Training runs for 5000 steps and produces a stable, expressive model despite the limited dataset.



7. Results

“PSM” captures the tone and structure of the original essays. Generated text tends to follow the author’s reflective style and maintains coherence across multiple sentences. The model learns to balance clarity, introspection, and progression of thought. It does not aim for encyclopedic knowledge. Instead, it behaves as a compressed representation of the writer’s internal reasoning style.

8. Intended Use

“PSM” is intended for:

- personal cognitive experimentation
- reflective dialogue
- single author creative writing
- style analysis
- academic exploration of small model behaviour

The system is narrow by design and is not intended to replace general purpose models.

9. Limitations

“PSM” inherits all limitations of its dataset. It has no external factual grounding. It cannot generalize beyond the structure of the essays. Its knowledge domain is bounded by the author. These constraints align with the purpose of the model and are not considered defects.

10. Conclusion

“PSM” demonstrates that meaningful language behaviour can emerge from a very small dataset when the model is designed for stability and when the dataset has strong internal structure. It shows that personal language modelling is possible at small scale and that depth of pattern matters more than raw volume. The model stands as a compact example of cognitive pattern synthesis driven by a single author’s worldview.

11. References

[1] Vaswani, A., Shazeer, N., Parmar, N., et al. Attention Is All You Need. Proceedings of the 31st Conference on Neural Information Processing Systems. 2017.

[2] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre Training. OpenAI Technical Report. 2018.

[3] Shazeer, N. GLU Variants Improve Transformer. Google Research Paper. 2020.

[4] Kingma, D. and Ba, J. Adam. A Method for Stochastic Optimization. arXiv. 2014.

[5] Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. arXiv. 2017.

[6] Karpathy, A. nanoGPT. Minimalistic Character Level Transformer Training Framework. GitHub. 2023.

[7] Phonex Legend. Personal Essays and Reflections Dataset. Private Manuscript. 2025.

[8] Ba, J., Kiros, J., and Hinton, G. Layer Normalization. arXiv. 2016.

[9] Smith, L. Cyclical Learning Rates for Training Neural Networks. IEEE Winter Conference on Applications of Computer Vision. 2017.