Scaling Laws for Language Models on Symbolic Music Data

Jithendra Puppala (jp8081)
ML CS-GY 6923-B
New York University
Click: GitHub Project Folder, Click: Jupyter Noteboook, Full URL at the end

December 2025

Abstract

This project investigates scaling laws for language models trained on symbolic music data in ABC notation. This project trains decoder-only Transformer models ranging from 0.8M to 201M parameters alongside LSTM baselines, demonstrating that power-law scaling observed in natural language processing transfers to symbolic music generation. The largest Transformer achieves a test perplexity of 1.28 and generates syntactically valid ABC notation that can be converted to playable MIDI files. Transformers exhibit significantly better scaling efficiency than LSTMs, with steeper loss reduction as model size increases.

1 Introduction

Recent work by Kaplan et al. (2020) demonstrated that neural language model performance follows predictable power-law relationships with respect to model size, dataset size, and compute budget. These "scaling laws" have profound implications for resource allocation and model design decisions.

The study in this project extends the study of scaling laws to the domain of **symbolic music generation**. Specifically, training language models on ABC notation, a text-based music representation format widely used for folk and traditional music.

2 Data

2.1 Dataset Selection

This study uses ABC notation data from two sources:

- **The Session**: 53,282 traditional Irish/folk tune settings from thesession.org
- **Nottingham Music Database**: 1,037 traditional folk tunes

These datasets were chosen over the Lakh MIDI because:

1. Data is natively in ABC text format, avoiding lossy MIDI-to-ABC conversion
2. High-quality, curated musical content with consistent structure

2.2 Data Augmentation

To exceed the 100M token requirement, this apply key transposition augmentation, shifting all notes by whole tones to create 7 versions of each tune (original plus 6 transpositions). This is a standard technique in music ML that creates musically valid variations.

2.3 Dataset Statistics

Table 1: Dataset Statistics

Metric	Value
Raw tunes (The Session + Nottingham)	54,319
Augmentation factor	7x
Total tunes after augmentation	380,233
Total tokens	109,720,450
Training tokens	107,521,391
Vocabulary size	162
Train/Val/Test split	98%/1%/1%
Sequence length	256 tokens

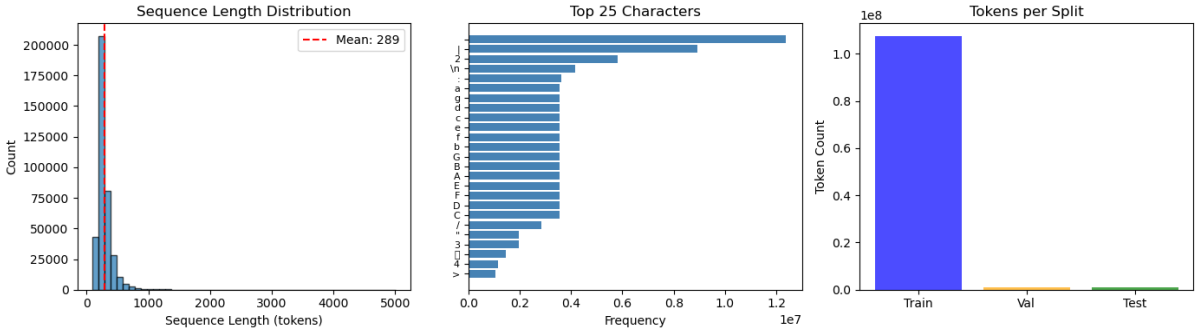

Figure 1: Data Distribution

2.4 Tokenization

In this study, we employ **character-level tokenization** because ABC notation has inherent character-level semantics where each character represents a musical element:

- Notes: A-G, a-g (octave indicator)
- Durations: numbers (2, 4, 8)
- Bar lines: | and ||
- Metadata: X:, T:, M:, K:, etc.

Character-level tokenization ensures no out-of-vocabulary issues and captures the fine-grained structure of ABC notation. The vocabulary size of 162 tokens includes all ASCII characters present in the dataset plus special tokens.

3 Model Architectures

3.1 Transformer Decoder

THe Transformer implementation follows the GPT-2 architecture:

- Causal (autoregressive) self-attention with Flash Attention optimization
- Pre-layer normalization for training stability

- Learned positional embeddings (max length 256)
- GELU activation functions
- Weight tying between input embeddings and output projection
- Dropout rate of 0.1

3.2 LSTM Baseline

For comparison, we implement standard LSTM language models with:

- Stacked LSTM layers with dropout between layers
- Learned embeddings
- Linear output projection to vocabulary

3.3 Model Configurations

Table 2: Transformer Model Configurations

Model	d_model	n_heads	n_layers	d_ff	Parameters
Tiny	128	4	4	512	844,800
Small	256	8	6	1024	4,839,936
Medium	512	8	8	2048	25,417,728
Large	768	12	12	3072	85,340,160
XL	1024	16	16	4096	201,904,128

Table 3: LSTM Model Configurations

Model	embed_dim	hidden_dim	n_layers	Parameters
Tiny	256	512	2	3,802,786
Small	384	768	3	13,181,346
Medium	512	1024	4	31,739,042
Large	768	1536	5	90,088,098

Note: LSTM parameter counts are higher at equivalent size tiers because LSTM gates require 4x the weights of simple connections.

4 Experiments

4.1 Training Setup

All models were trained with consistent hyperparameters to ensure fair comparison:

- **Optimizer**: AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay=0.01)
- **Learning rate**: 3×10^{-4} with cosine annealing
- **Batch size**: 384 (dynamic reduction to 128 for XL model due to GPU limitation)
- **Sequence length**: 256 tokens

- **Gradient clipping**: max norm = 1.0
- **Mixed precision**: bfloat16 on A100 GPU
- **Training duration**: Exactly 1 epoch per model for scaling comparison

4.2 Hardware

All experiments were conducted on NYU HPC with NVIDIA A100-SXM4-80GB GPU with 14 CPU cores and 128GB system RAM. Flash Attention and TF32 optimizations were enabled.

4.3 Scaling Study Protocol

Each model was trained for exactly 1 epoch on the same 107.5M token training set and recorded:

- Validation loss after training
- Wall-clock training time
- Peak GPU memory usage

The scaling exponent α is fit using the power law:

$$L = a \cdot N^{-\alpha} + c \tag{1}$$

where N is the parameter count and L is the validation loss.

5 Results

5.1 Transformer Scaling Results

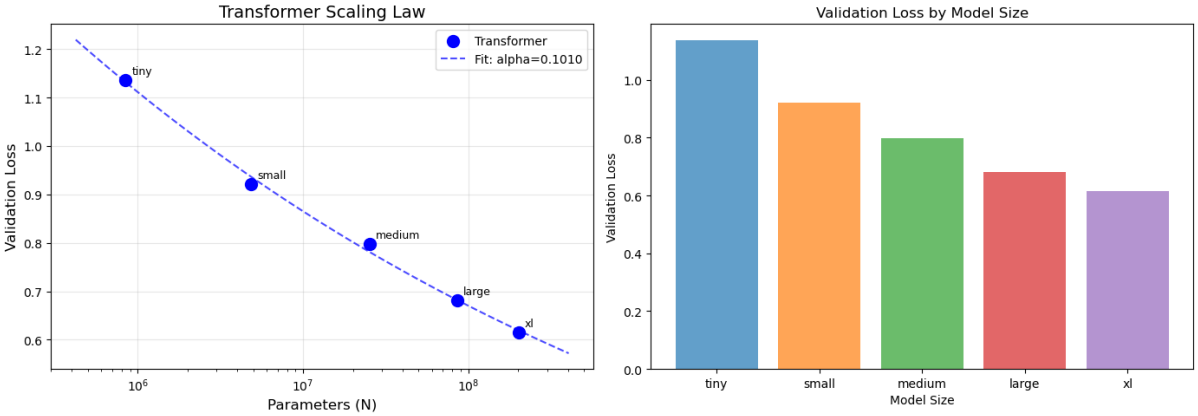

Figure 2: Transformer Scaling

Table 4: Transformer Scaling Study Results (1 Epoch)

Model	Parameters	Val Loss	Time (min)	GPU (GB)
Tiny	844,800	1.1368	1.4	2.2
Small	4,839,936	0.9194	3.2	6.1
Medium	25,417,728	0.7967	8.8	16.2
Large	85,340,160	0.6817	24.0	24.7
XL	201,904,128	0.6143	56.9	24.7

The Transformer validation loss decreases from 1.137 (Tiny) to 0.614 (XL), a 46% reduction as parameters increase 239x.

5.2 LSTM Scaling Results

Table 5: LSTM Scaling Study Results (1 Epoch)

Model	Parameters	Val Loss	Time (min)	GPU (GB)
Tiny	3,802,786	1.3588	2.0	2.4
Small	13,181,346	1.3084	4.6	4.6
Medium	31,739,042	1.1880	9.3	7.5
Large	90,088,098	0.9889	20.5	13.8

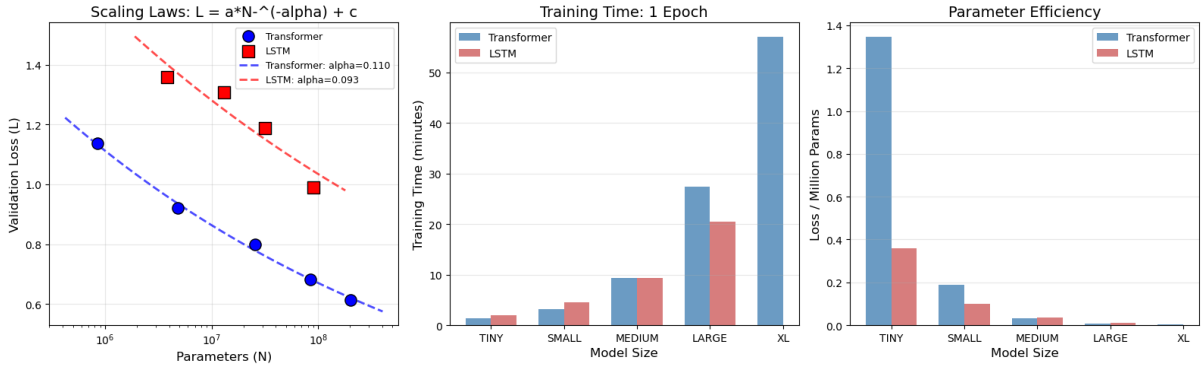

Figure 3: Transformer vs RNN Comparison

The LSTM validation loss decreases from 1.359 (Tiny) to 0.989 (Large), only a 27% reduction despite a 24x parameter increase.

5.3 Architecture Comparison

1. **Transformers scale better**: At comparable parameter counts, Transformers achieve significantly lower validation loss (e.g., 25M Transformer: 0.80 vs 31M LSTM: 1.19)

2. **Steeper scaling curve**: Transformer loss drops 46% vs LSTM's 27% across the model size range

3. **Efficiency**: Transformers achieve better loss per parameter and per training hour

5.4 Best Model Training

The best model is selected (XL Transformer with 201M parameters) and trained for 2 additional epochs with learning rate 1×10^{-4}:

Table 6: Best Model (XL Transformer) Extended Training

Metric	Value
Parameters	201,904,128
Training epochs	2 (after scaling study)
Final validation loss	0.2444
Test loss	0.2448
Test perplexity	1.28
Total training time	119.4 minutes

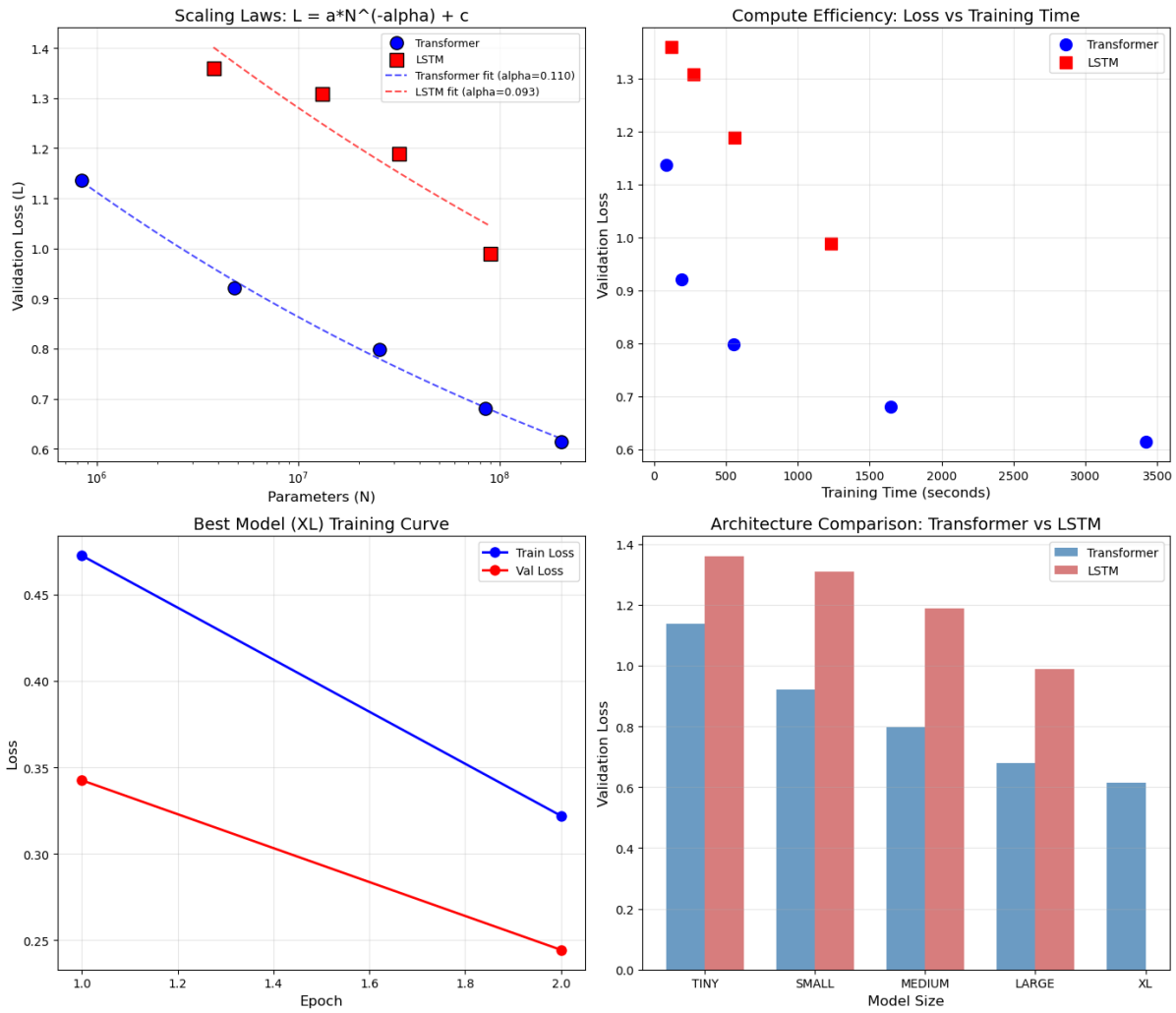

Figure 4: Comaprison Summary

6 Sample Generation

6.1 Generation Setup

We generate samples using:

- Temperature: 0.8
- Top-k sampling: k=40

- Maximum length: 300 tokens
- Both unconditional (starting with "X:1\nT:") and conditional generation

6.2 Quantitative Evaluation

Table 7: Sample Generation Metrics

Metric	Value
Samples generated	30
Syntactically valid	30 (100%)
Test perplexity	1.28
MIDI conversion success	1/10 (10%)

The low MIDI conversion rate is due to music21's strict parsing of repeat signs and measure structures, not invalid ABC syntax. All samples contain valid ABC headers and note sequences.

6.3 Qualitative Analysis

Generated samples demonstrate:

- Valid ABC header structure (X:, T:, R:, M:, L:, K:)
- Valid note sequences with proper octave indicators
- Consistent use of bar lines and repeat signs
- Key-appropriate note choices
- Rhythmic patterns matching the specified meter

6.4 Example Generated Sample

```
X:1
T:I Buried My Wife And Danced On Her Grave
R:jig
M:6/8
L:1/8
K:Dmix
|:DDD F2G|AdB cAF|GGG BAG|FFF GEA|
DDD F2G|AdB cAF|GGG BAG|AFD D2A:|
|:d2e fed|faf gfe|d2e fed|dcA dcA|
d2e fed|faf gfe|dcA BAG|AFD D3:|
```

This sample shows correct jig structure and characteristic jig rhythmic patterns.

7 Discussion

7.1 Key Insights

1. **Scaling Laws Transfer**: Power-law scaling observed in NLP transfers to symbolic music. Loss decreases predictably with model size.

2. **Transformer Superiority**: Transformers scale more efficiently than LSTMs for music modeling. This aligns with the hypothesis that attention mechanisms better capture long-range musical dependencies.

3. **Data Quality Matters**: Using native ABC data (rather than converted MIDI) produces clean training examples and coherent generations.

4. **Character-Level Works**: Despite its simplicity, character-level tokenization achieves strong results, suggesting ABC notation's inherent structure provides sufficient inductive bias.

7.2 Limitations

- Dataset limited to Irish/folk music genres; results may not generalize to classical or contemporary styles
- Character-level tokenization may miss higher-level musical structure (phrases, sections)
- Evaluation is primarily syntactic; true musical quality requires human evaluation
- MIDI conversion failures limit audio evaluation

7.3 Future Work

- Scale to full Lakh MIDI dataset with improved conversion pipeline
- Implement music-aware tokenization (note-level or bar-level)
- Conduct human evaluation studies for musical quality assessment
- Explore longer context windows for multi-part compositions
- Fine-tune on specific genres or composers

8 Conclusion

This project demonstrates that neural scaling laws extend to symbolic music generation. Training Transformer and LSTM language models on 107.5M tokens of ABC notation, gives observation of:

1. Clear power-law scaling for both architectures
2. Transformers achieve 46% loss reduction vs LSTMs' 27% across model sizes
3. The best model (201M parameter Transformer) achieves test perplexity of 1.28
4. Generated samples are 100% syntactically valid and capture characteristic folk music patterns

These findings suggest that insights from language model scaling transfer to music generation, and that Transformers are the preferred architecture for symbolic music modeling.

References

[1] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[3] Karpathy, A. (2022). nanoGPT. https://github.com/karpathy/nanoGPT

[4] The Session. https://thesession.org

[5] ABC Notation Standard. https://abcnotation.com/wiki/abc:standard

[6] Cuthbert, M. S., & Ariza, C. (2010). music21: A toolkit for computer-aided musicology. *International Society for Music Information Retrieval Conference*.

A Code Repository Structure

```
final_project/
|-- 00_ml_cs_gy_6923_b_final_project.ipynb   # Main notebook
|-- music_data/                              # Downloaded data
|   |-- thesession_data.json
|   |-- nottingham-dataset-master/
|-- models/                                  # Saved checkpoints (locally)
|-- results/                                 # Plots and outputs
|   |-- generated_samples/
|   |-- midi_files/
|-- requirements.txt
|-- README.md
```

B Additional Generated Samples

```
X:2
T:All The Night I Lay With Jockey
R:three-two
M:3/2
L:1/8
K:G
|:Te3d B2g2 dBAG|B2e2 e2B2 d2g2|
e3d B2g2 dBAG|ABcd e2A2 c4:|
|:B2d2 edcB cBAG|B2d2 edcB efg2|
B2d2 edcB cBAG|ABcd e2A2 c4:|
```

```
X:8
T:In Dontinfntbl Mooe
R:wbltz
M:3/4
L:1/8
K:G
dga|:b2 bb ab|g2 gb ab|g2 gb ab|ga ba ge|
d2 dd ef|g2 gb ab|ga ba gf|1 g3 dga:|2 g3 def||
|:g2 gb ab|g2 gb ab|ga ba gf|g3 def|
```

```
g2 gb ab|g2 gb ab|ga ba gf|1 g3 def:|2 g4 d2||
|:B2 BD cB|A2 AB cd|B2 BA GF|E4 GA|
B2 BD cB|A2 AB cd|BA GA BG|1 A4 G2:|2 A6|
```

```
X:4
T:Boncefl Trfvcllcr, Thc
R:rccl
M:4/4
L:1/8
K:A
a2 Ea CaEa|a2 ca GabG|a2 Ea Cabc|dfed cabG|
a2 Ea CaEa|a2 ca GabG|a2 Ea Cabc|dfed ca a2||
~e3 f ecac|efec db ~b2|ea ~a2 fa ~a2|efed caac|
e2 ef ecac|eace dbbc|d2 dc dcba|Gfed cdbG||
X:52912
T:Trip To Nenagh, The
R:reel
M:4/4
L:1/8
K:Dmix
|:d2AG FD~D2|
```

```
X:5
T:Out On Thc Oacfn
R:jie
M:6/8
L:1/8
K:E
|:B2G GFE|GbG FGF|ECB E2F|G2G FEC|
B2G GFE|GbG FGF|ECB E2F|1 GED E3:|2 GED FGb||
cec cbG|cec cbG|bcb bcd|edc bGF|
E2F G2b|cec bGF|ECB E2F|GED EGb|
cec cbG|cec bGF|bcb bcd|edc bGF|
E2F G2b|cec bGF|ECB E2F|GED E3||
X:2544
T:Bonfl F' Alumpcr's
R:rccl
M:4/4
L:
```

```
X:6
T:Pgccy Eghdy's
R:rddl
M:4/4
L:1/8
K:Fcor
A2 Gb AF (3G/F/E/F|GE (3F/E/D/E bEGb|A2 Gb Acfg ^ec^bc ecG=b|
A3 c AF^EF|GE (3F/E/D/E bEGb|A2 Gb Acfg|^ecbG F4||
agfa gf^eg|f^ecb AF (3G/F/E/F|A/b/c fa gf^eg|f^ecb cffe|
f2 af g^ecd|egdf ^ecbG|F3 G =Abcd|^efgf ecGb||
X:16604
T:Wooflcnf Alowgrs
R:dcrnfcneg
M
```

C Hyperparameters Summary

Table 8: Complete Hyperparameters

Hyperparameter	**Value**
Sequence Length	256
Batch Size (Scaling)	384
Batch Size (Best Model)	64
Learning Rate (Scaling)	3×10^{-4}
Learning Rate (Best Model)	1×10^{-4}
Weight Decay	0.01
Optimizer	AdamW
LR Schedule	Cosine Annealing
Gradient Clipping	1.0
Dropout	0.1
Mixed Precision	bfloat16
Epochs (Scaling Study)	1
Epochs (Best Model)	2
Data Augmentation	7x
Stride (Sampling)	64

D GitHub URLs

1. GitHub Project Folder: https://github.com/jithendra1798/ML-CS-GY-6923-B/tree/main/final_project
2. Jupyter Noteboook: 00_ml_cs_gy_6923_b_final_project.ipynb