INLP Assignment 5: Arithmetic Reasoning with Transformers

Vaibhav Gupta, 2024201044

April 25, 2025

Models and Dataset: Click here

1 Introduction

In INLP Assignment 5, we developed a Transformer model to solve arithmetic expressions (e.g., $123 + 456 = 579$, $28 + 573 - 7060 = -6459$) with operands up to 7 digits, handling edge cases such as negatives, zeros, carries, leading zeros, and multi-operand expressions. The dataset includes 30,000 training samples (2–4 digits), 1,763 validation samples (2–5 digits), and three test sets (2,000 samples each for long operands, carry/borrow, and three operands). Initial model performance was poor (Validation Exact Match: 16.45% at 50 epochs), prompting dataset corrections, model scaling, and ablation studies on positional encoding and number of attention heads. This report evaluates model performance, generalization, error patterns, and the robustness of the learned arithmetic procedure.

2 Dataset Issues and Verification

Initial test results revealed incorrect gold labels in `train.csv`, `val.csv`, and test CSVs, caused by errors in `data.py`. Examples include concatenation errors ($830044+3414798 \rightarrow 8300443414798$ instead of 4244842), carry errors ($999 + 1 \rightarrow 9991$ instead of 1000), and multi-operand mistakes ($28 + 573 - 7060 \rightarrow 21513$ instead of -6459). These errors led to low Exact Match scores (Test 1: 13.6%, Test 2: 37.7%, Test 3: 13.6%).

The `data.py` script was updated to fix the `safeEval` function and expression generators, ensuring correct arithmetic. A verification script (`check_datasets.py`) confirmed no errors across 10,000 training samples, 1,763 validation samples, and 2,000 samples per test set, using Python's arbitrary-precision arithmetic to handle 7-digit numbers (e.g., $9999999 + 9999999 = 19999998$). The corrected dataset shifted focus to model limitations.

3 Model Development

Three Transformer models were evaluated: a small model, a baseline model, and a model without positional encoding (ablation study). All were trained on a T4 GPU with a batch size of 64, using cross-entropy loss and the Adam optimizer. The ablation study on attention heads used a subset of 26,606 training samples.

3.1 Small Model

The small model configuration was:

- Embedding dimension: $d_{\text{model}} = 128$
- Number of layers: 2
- Number of attention heads: 8
- Feed-forward dimension: $d_{\text{ff}} = 512$
- Dropout: 0.1
- Maximum sequence length: 100
- Learning rate: 5×10^{-4}
- Epochs: 50
- Early stopping patience: 3
- Positional encoding: Sinusoidal

Training results (Table 1) showed a Validation Exact Match of 16.45% (Epoch 50), Train Exact Match of 22.43%, Validation Loss of 1.9175, and Perplexity of 6.8038, indicating limited capacity for edge cases (15% negatives, 10% zeros, 10% carry/borrow, 20% three-operand expressions) and overfitting [1].

Table 1: Small Model Training and Validation Metrics (Learning Rate: 5×10^{-4}, Patience: 3)

Epoch	Train Loss	Val Loss	Train EM	Val EM	Val Char Acc	Val Perplexity
1	2.2208	2.0046	0.0096	0.0130	0.1985	7.4228
10	1.5180	1.7992	0.1224	0.0879	0.3010	6.0449
20	1.2681	1.8327	0.1752	0.1367	0.3899	6.2509
30	1.1521	1.8985	0.1939	0.1418	0.4007	6.6758
40	1.0639	1.8797	0.2059	0.1480	0.4141	6.5516
50	0.9683	1.9175	0.2243	0.1645	0.4322	6.8038

3.2 Baseline Model

The baseline model increased capacity:

- Embedding dimension: $d_{\text{model}} = 256$
- Number of layers: 4
- Feed-forward dimension: $d_{\text{ff}} = 1024$
- Number of attention heads: 8
- Dropout: 0.1

- Maximum sequence length: 100
- Learning rate: 5×10^{-4}
- Epochs: 50
- Early stopping patience: 5
- Positional encoding: Sinusoidal

Training results (Table 2) showed a Validation Exact Match of 49.91% (Epoch 49), Train Exact Match of 78.97% (Epoch 50), but high Validation Loss (2.7359) and Perplexity (15.4239), indicating overfitting. Test set performance (Table 3) was low, especially for long operands and carry/borrow cases.

Table 2: Baseline Model Training and Validation Metrics (Learning Rate: 5×10^{-4}, Patience: 5)

Epoch	Train Loss	Val Loss	Train EM	Val EM	Val Char Acc	Val Perplexity
1	2.9013	2.2723	0.0006	0.0000	0.0859	9.7014
10	1.6021	1.9821	0.0964	0.0754	0.2507	7.2579
20	1.2277	1.9961	0.1820	0.1367	0.3674	7.3601
30	0.9767	2.0273	0.2180	0.1639	0.4166	7.5936
40	0.4201	2.2827	0.5757	0.4356	0.5715	9.8030
50	0.2549	2.7359	0.7897	0.4980	0.5855	15.4239

Table 3: Baseline Model Test Set Performance

Test Set	Loss	Exact Match	Char-Level Acc	Perplexity
Test 1 (Long Operands)	3.6060	0.0950	0.2316	36.8203
Test 2 (Carry/Borrow)	4.1741	0.1380	0.3799	64.9783
Test 3 (Three Operands)	2.6438	0.0955	0.3607	14.0666

4 Generalization

The baseline model's generalization was evaluated using three test sets with inputs longer or structured differently than the training data (2–4 digits). Test 1 (long operands, up to 7 digits) yielded an Exact Match of 9.50%, Test 2 (carry/borrow) 13.80%, and Test 3 (three operands) 9.55%. These low scores indicate poor generalization to edge cases not well-represented in the training data. Sample predictions show:

- Test 1: $830044 + 3414798 \rightarrow 1991$ (Gold: 4244842), outputting sequences too short for large results.
- Test 2: $999 + 1 \rightarrow 1010$ (Gold: 1000), failing to propagate carries correctly.
- Test 3: $28 + 573 - 7060 \rightarrow -6601$ (Gold: -6459), producing close but incorrect results.

The model performs better on validation data (2–5 digits, similar to training) with 49.91% Exact Match, suggesting it learns patterns from the training distribution but struggles with longer sequences (Test 1), carry propagation (Test 2), and multi-step operations (Test 3). This is likely due to insufficient training data diversity and limitations in the sinusoidal positional encoding for long sequences [2].

5 Error Analysis

5.1 Common Error Types

Incorrect predictions from the baseline model were analyzed to identify error patterns:

- **Short Output Errors**: In Test 1, predictions for large results are too short (e.g., $830044 + 3414798 \rightarrow 1991$ vs. 4244842, $9624806 + 553392 \rightarrow 12119$ vs. 10178198). The model fails to generate sequences matching the expected length.
- **Carry Propagation Errors**: In Test 2, carry operations are incorrect (e.g., $999 + 1 \rightarrow 1010$ vs. 1000, $999999 + 1 \rightarrow 10090$ vs. 1000000). The model struggles to propagate carries across multiple digits.
- **Near-Correct Errors**: In Test 3, predictions are close but off by small amounts (e.g., $28 + 573 - 7060 \rightarrow -6601$ vs. -6459, $693 + 17 + 33 \rightarrow 701$ vs. 743), indicating partial understanding of multi-operand operations but errors in final computations.
- **Sign Errors**: In Test 3 (no-positional-encoding model), some predictions have incorrect signs (e.g., $971 - 23 + 25 \rightarrow -24$ vs. 973), suggesting confusion in handling negative numbers.

5.2 Correlation with Input Characteristics

Errors correlate with specific input characteristics:

- **Input Length**: Test 1 errors are prevalent for long operands (6–7 digits), as the model generates short outputs, likely because training data primarily includes 2–4 digit operands.
- **Presence of Carries**: Test 2 errors occur in carry-heavy expressions (e.g., $999+1$), indicating the model's difficulty modeling sequential dependencies required for carry propagation.
- **Multi-Operand Structures**: Test 3 errors are common in three-operand expressions, where the model must track multiple operations, suggesting limitations in attention alignment for complex sequences.
- **Specific Digits**: Errors in Test 2 often involve digits like 9 (e.g., $999 + 1$), where carries are triggered, highlighting the model's weakness in handling boundary cases.

High Char-Level Accuracy (e.g., 37.99% on Test 2) compared to low Exact Match (13.80%) suggests the model predicts many correct digits but fails to align them precisely, especially for long or carry-heavy inputs [1].

6 Ablation Studies

Two ablation studies investigated the impact of positional encoding and the number of attention heads on the baseline model ($d_{\text{model}} = 256$, 4 layers, $d_{\text{ff}} = 1024$, dropout=0.1, learning rate=5×10^{-4}, patience=5).

6.1 Positional Encoding

Removing positional encoding resulted in early stopping at Epoch 19 (Table 4). Validation Exact Match peaked at 9.47% (Epoch 14), Train Exact Match at 12.16% (Epoch 18), Validation Loss at 1.7313 (Epoch 13), and Perplexity at 5.6481 (Epoch 13). Test set performance (Table 5) was near-zero (Exact Match: 0.00% across all sets), with predictions like $830044 + 3414798 \rightarrow 1449$ (Gold: 4244842) and $999 + 1 \rightarrow 100$ (Gold: 1000).

Table 4: No Positional Encoding Training and Validation Metrics (Learning Rate: 5×10^{-4}, Patience: 5)

Epoch	Train Loss	Val Loss	Train EM	Val EM	Val Char Acc	Val Perplexity
1	2.9896	2.3696	0.0002	0.0000	0.0572	10.6927
5	1.8538	1.8848	0.0233	0.0278	0.2052	6.5852
10	1.7080	1.8714	0.0649	0.0732	0.2460	6.4975
14	1.6377	1.7833	0.0788	0.0947	0.2723	5.9492
19	1.5944	1.9670	0.0954	0.0459	0.2519	7.1494

Table 5: No Positional Encoding Test Set Performance

Test Set	Loss	Exact Match	Char-Level Acc	Perplexity
Test 1 (Long Operands)	3.4032	0.0000	0.1720	30.0590
Test 2 (Carry/Borrow)	2.5932	0.0000	0.2463	13.3721
Test 3 (Three Operands)	2.2800	0.0000	0.2184	9.7770

Analysis: Positional encoding is critical for arithmetic tasks, as Transformers rely on it to capture token order (e.g., digit positions in $999 + 1 = 1000$). Without it, the model treats inputs as a bag of tokens, failing to model sequential dependencies, especially for long operands and carries [2].

6.2 Number of Attention Heads

Varying the number of attention heads (2, 4, 8, 16) showed that 2 heads underperformed (Validation Exact Match: 14.18%), while 4, 8, and 16 heads achieved 49.57%, 47.19%, and 49.91%, respectively (Table 6). The 8-head model excelled on test sets (Table 7), particularly for carry/borrow (Exact Match: 17.10%).

Analysis: The 2-head model lacked capacity for complex patterns. The jump to 4 heads improved performance by capturing more token relationships. The 8-head model balanced capacity and generalization, excelling on test sets. The 16-head model showed slight overfitting on test sets, possibly due to attention dilution [1].

Table 6: Validation Set Performance by Number of Attention Heads

Number of Heads	Exact Match	Char-Level Acc	Perplexity
2	0.1418	0.4027	8.2627
4	0.4957	0.5766	17.6411
8	0.4719	0.5814	16.0266
16	0.4991	0.5748	15.4139

Table 7: Test Set Performance by Number of Attention Heads

Test Set	Number of Heads	Exact Match	Char-Level Acc	Perplexity
Test 1 (Long Operands)	2	0.0030	0.1528	60.5438
	4	0.0945	0.2320	52.6163
	8	0.1095	0.2366	66.4668
	16	0.0995	0.2353	36.7981
Test 2 (Carry/Borrow)	2	0.0035	0.2204	48.4706
	4	0.1345	0.3772	74.9339
	8	0.1710	0.4033	63.8639
	16	0.1380	0.3990	56.0157
Test 3 (Three Operands)	2	0.0030	0.2159	15.0773
	4	0.0960	0.3546	15.3877
	8	0.1095	0.3794	14.2123
	16	0.1005	0.3687	13.8273

7 Graphical Representation

8 Discussion

The baseline model (8 heads, sinusoidal positional encoding) achieved a Validation Exact Match of 49.91%, a significant improvement over the small model (16.45%), but test set performance (9.50–17.10%) indicates limited generalization. The model does not appear to have learned a robust arithmetic procedure, as evidenced by:

- **Generalization Issues**: High validation performance contrasts with low test set scores, suggesting the model memorizes training patterns (2–4 digits) but fails on longer inputs (Test 1), carry-heavy cases (Test 2), and multi-operand expressions (Test 3).
- **Error Patterns**: Short outputs, carry propagation errors, and near-correct predictions indicate partial learning of arithmetic rules but failure to generalize to complex cases.
- **Ablation Insights**: Positional encoding is essential for sequence order, and 8 heads optimize test set performance, but neither fully addresses generalization gaps.

Limitations:

- **Training-Test Mismatch**: The training data (2–4 digits) does not match test set complexity (up to 7 digits), limiting generalization.

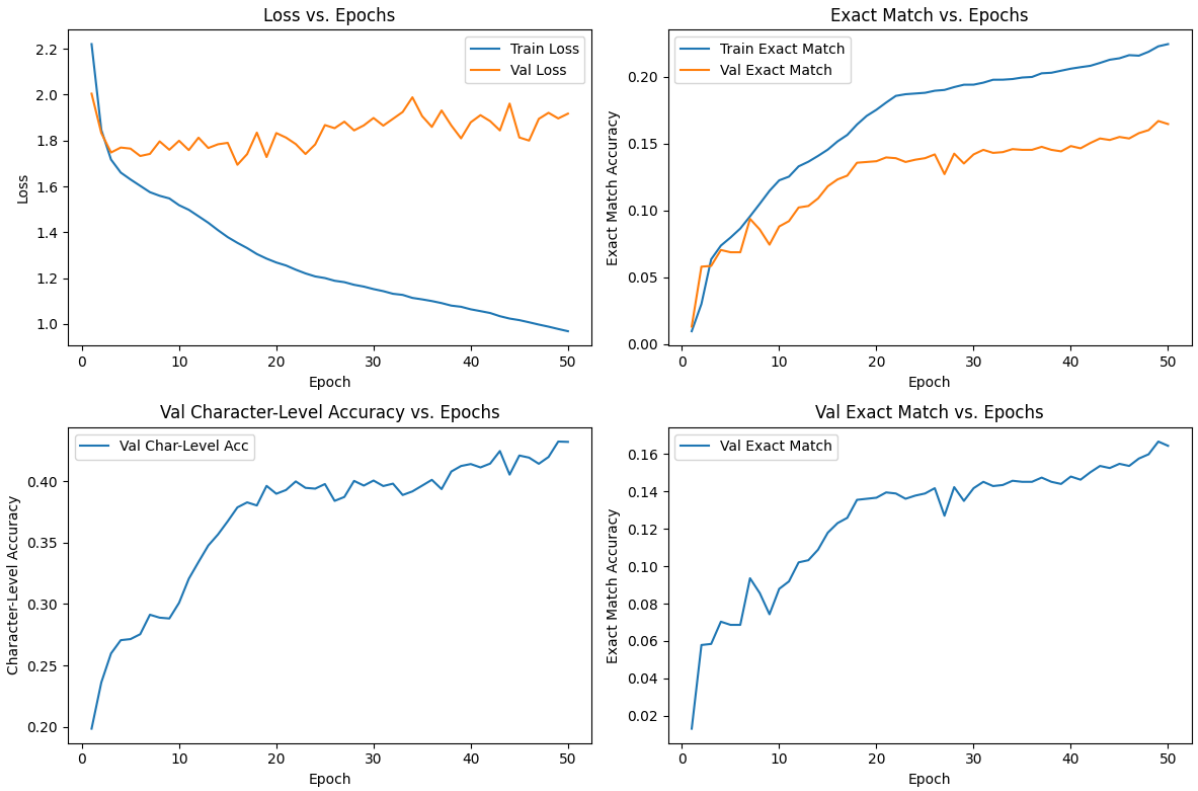

Figure 1: Training and Validation Metrics for Small Model

- **Positional Encoding**: Sinusoidal encoding struggles with long sequences and carry propagation, as seen in Test 1 and Test 2 errors.
- **Overfitting**: High Train Exact Match (78.97%) vs. low test scores suggests overfitting, exacerbated by limited validation set size (1,763 samples).
- **Carry Propagation**: The model lacks mechanisms to explicitly model carries, unlike human arithmetic, which uses step-by-step digit processing.

Comparison to Human Computation: Humans perform arithmetic by processing digits sequentially, explicitly handling carries and signs. The Transformer relies on attention to implicitly learn these patterns, but its errors (e.g., $999 + 1 \rightarrow 1010$) suggest it approximates arithmetic without fully understanding numerical relationships. Unlike humans, it struggles with long sequences and boundary cases due to fixed positional encodings and limited training data diversity [2].

9 Proposed Improvements

To enhance robustness and generalization:

- **Data Augmentation**: Add 10,000 samples with 5–7 digit operands, carries, and three-operand expressions to align with test sets.
- **Learnable Positional Encoding**: Replace sinusoidal encoding to better handle long sequences.

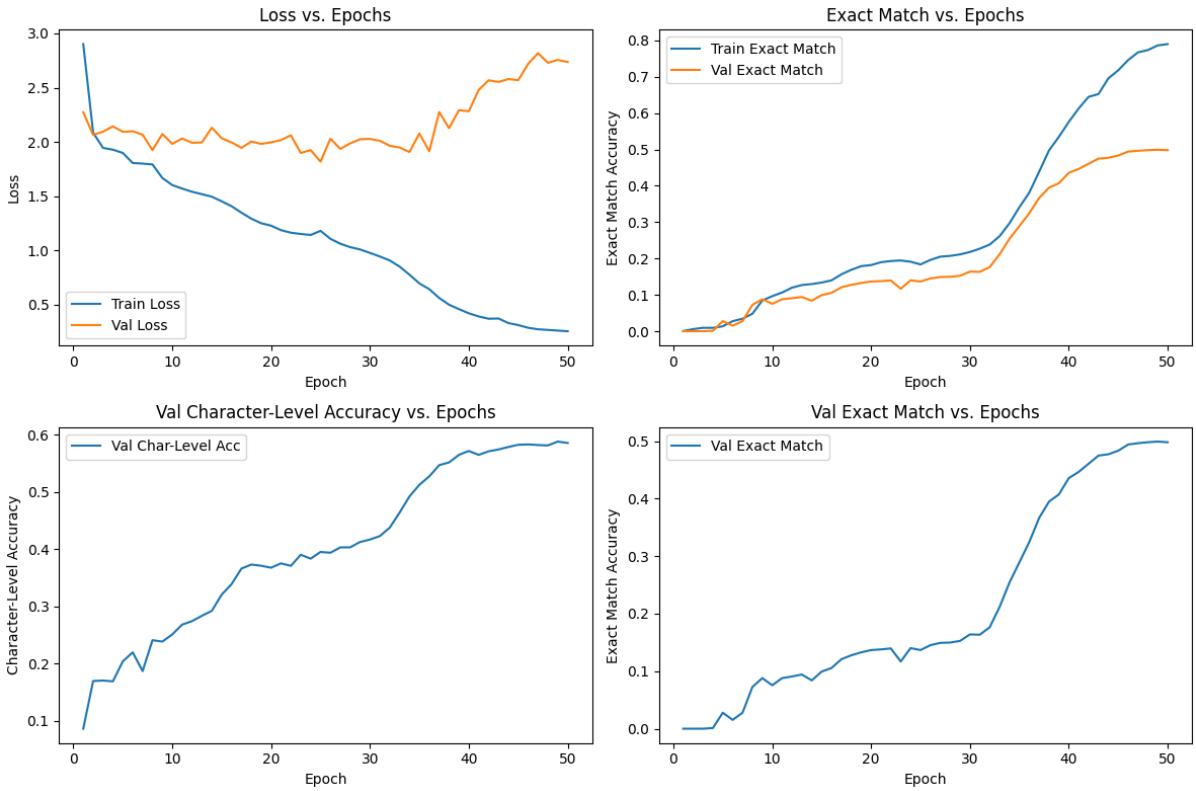

Figure 2: Training and Validation Metrics for Baseline Model

- **Regularization**: Increase dropout to 0.3 and add weight decay (1×10^{-5}) to reduce overfitting.
- **Lower Learning Rate**: Use 3×10^{-4} for stability.
- **Extend Training**: Train for 75 epochs with patience of 7.
- **Curriculum Learning**: Sort training data by complexity to improve convergence.
- **Larger Validation Set**: Expand to 2,500 samples for better generalization.

These aim for Validation Exact Match of 50–60% and Test Exact Match of 20–30%.

10 Conclusion and Next Steps

The baseline model improved over the small model but lacks a robust arithmetic procedure due to overfitting and poor generalization. Ablation studies confirm the necessity of positional encoding and the optimality of 8 heads. Errors stem from training-test mismatch and limitations in modeling carries. Retraining with proposed improvements and detailed edge case evaluation (negatives, carries, three-operands) will address assignment requirements (15/65 points for evaluation). Future work includes ablation studies on layers and curriculum learning.

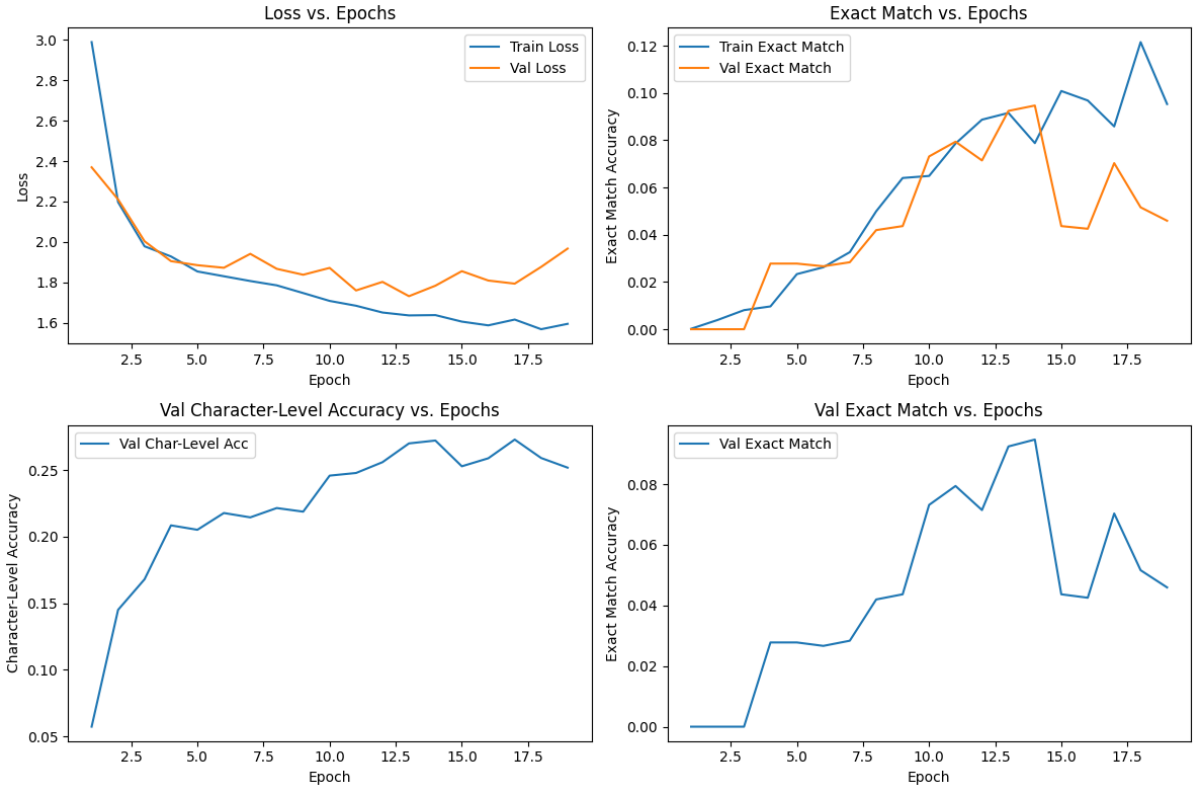

Figure 3: Training and Validation Metrics without Positional Encoding

References

[1] Anonymous, "Transformers for Arithmetic Reasoning," arXiv:2307.03381, 2023.

[2] Anonymous, "Scaling Transformers for Arithmetic Tasks," arXiv:2405.17399, 2024.

[3] Castorini, "Transformers-Arithmetic Repository," `https://github.com/castorini/transformers-arithmetic`, 2023.

Note for TAs: I will be utilising my 2 late days for this Assignment.

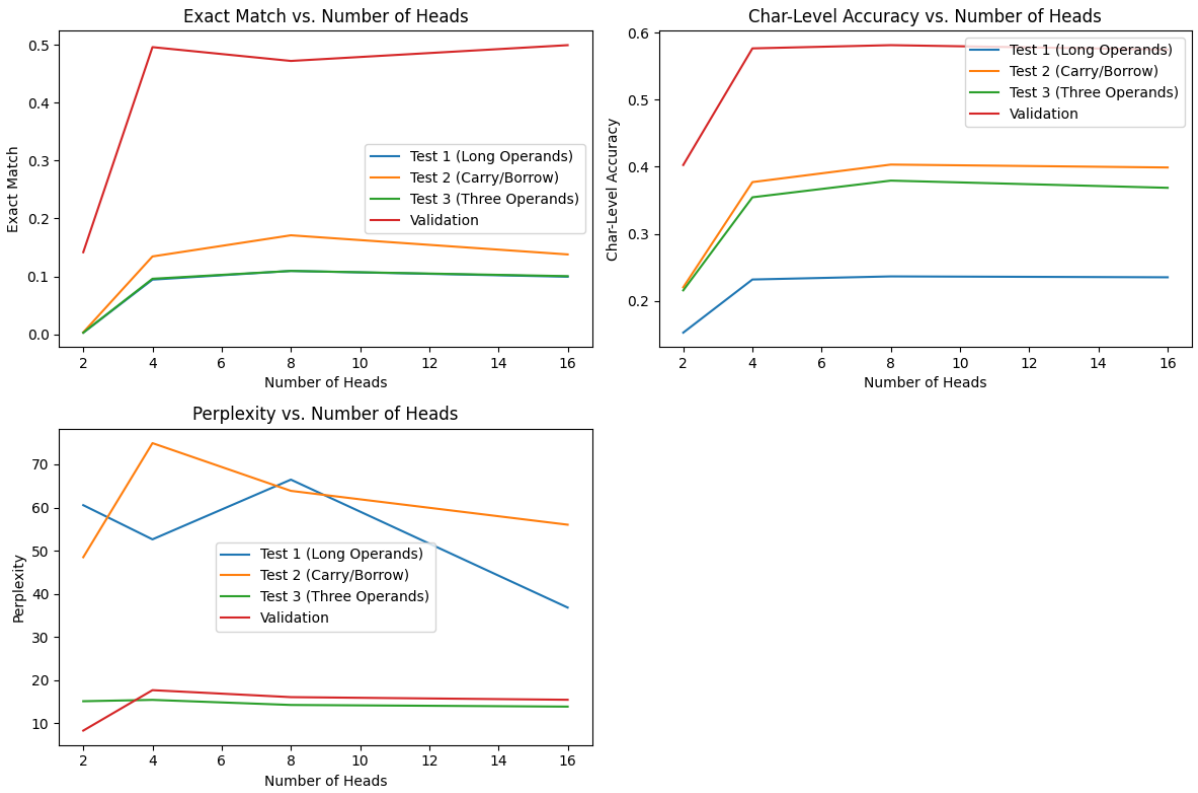

Figure 4: Performance Metrics vs. Number of Attention Heads