ANLP Project Mid-Submission Report: Data Preprocessing and Fine-tuning for AI Visual Novel Generation

Jatin Agrawal
2023114014

Amisha
2023114003

Aaryan Kashyap
2023114006

1 Project Overview

The code for the project so far is available at: Github Link.

Visual novels (VNs) are interactive story-based games that combine branching narratives with visual and audio elements such as background art, character sprites, and music to create an immersive storytelling experience.

Traditional VNs often suffer from limited replayability because of static story branches, while current AI-generated stories struggle with coherence and lack meaningful human authorial control. Our project bridges this gap by designing a **hybrid narrative system** that combines human creative oversight with AI-driven dynamic content generation.

Primary Objectives

1. A fine-tuned generation model for structured narrative output.
2. A benchmark for evaluating creative quality.
3. An ablation study to identify factors that improve creative output.
4. Create a web-based visual novel engine which can convert the generated json script to an interactive visual novel

2 Background and Related Work

2.1 Screenplay Understanding and Annotation

The task of converting unstructured screenplay text into structured, machine-readable formats has been explored in various domains. Liu et al. (2020) introduced ScriptBase, demonstrating automated screenplay parsing techniques for scene boundary detection and character identification. Similarly, Walker et al. (2011) developed the Scheherazade system for narrative generation, which utilized structured story representations similar to our annotation schema.

Our annotation format draws inspiration from visual novel engines like Ren'Py[1], which uses structured scene descriptions with character positioning, expressions, and dialogue flow. The concept of branching narratives with state management is fundamental to interactive storytelling systems (Riedl and Bulitko, 2013).

2.2 Large Language Models for Annotation

Recent advances in Large Language Models (LLMs) have enabled automated annotation tasks. Brown et al. (2020) demonstrated few-shot learning capabilities with GPT-3, showing that models can adapt to specific formats with minimal examples. Wei et al. (2022) introduced chain-of-thought prompting, which influenced our iterative prompt refinement strategy.

For local deployment, we utilized Ollama[2], an open-source framework for running quantized LLMs locally. The Llama 3.2:1b model (Touvron et al., 2023) provided lightweight inference capabilities, though with accuracy trade-offs compared to larger models.

Google's Gemini 2.5 Flash[3] offered superior performance with extended context windows (up to 1M tokens), making it suitable for processing lengthy screenplay documents while maintaining annotation consistency.

2.3 Fine-tuning Methodologies

Parameter-Efficient Fine-Tuning (PEFT) techniques, particularly LoRA (Low-Rank Adaptation) (Hu et al., 2021), have revolutionized model fine-tuning by reducing trainable parameters from

[1] https://www.renpy.org
[2] https://github.com/ollama/ollama
[3] https://deepmind.google/technologies/gemini

billions to millions while maintaining performance. LoRA adds trainable rank decomposition matrices to attention layers, achieving comparable results to full fine-tuning with significantly lower memory requirements.

QLoRA (Dettmers et al., 2023) extends this by combining LoRA with 4-bit quantization using the NormalFloat4 (NF4) data type, enabling fine-tuning of large models on consumer hardware. Our implementation leverages these techniques through the BitsAndBytes library and Hugging Face PEFT framework.

Google's Gemma model family (Gemma Team2024 , 2024) provides open-weights models specifically designed for instruction-following tasks, making them ideal for domain-specific fine-tuning in creative writing applications.

2.4 Decoding Strategies for Text Generation

Generating coherent and creative text from a language model is not solely dependent on the model's training; the decoding strategy used during inference plays a critical role. Decoding strategies are algorithms that select the next token from the probability distribution generated by the model at each step. Early and common approaches include deterministic methods like **Greedy Search**, which simply selects the most probable token, and **Beam Search**, which expands upon this by keeping track of several of the most likely sequences (Sutskever et al., 2014). While effective for tasks requiring factual accuracy like translation, these methods often result in text that is deterministic, repetitive, and lacks diversity, a phenomenon termed "text degeneration" by Holtzman et al. (2020).

To counteract this, stochastic sampling methods were introduced to add variability. The most fundamental of these is **Temperature Sampling**, which rescales the model's logits before the softmax function to either sharpen or flatten the probability distribution (Hinton et al., 2015). While this introduces valuable randomness, it can be difficult to tune and may produce incoherent text by over-sampling from the long tail of the distribution. Building on this, Fan et al. (2018) proposed **Top-K Sampling**, which truncates the vocabulary to the K most likely tokens before sampling, effectively eliminating highly improbable tokens.

The current state-of-the-art approach, which provides a more dynamic solution, is **Nucleus Sampling** or **Top-P** (Holtzman et al., 2020). Instead of being limited to a fixed number K, Top-P selects the smallest set of tokens whose cumulative probability exceeds a threshold p. This allows the sampling pool to be large when the model is uncertain (e.g., at the start of a sentence) and small when the next token is highly predictable. This adaptability has been shown to produce text that is both more coherent and more creative than previous methods, making it a vital technique for creative generation tasks like ours.

3 Proposed Methodology

1. **Manual Annotation:** Annotate a subset of horror movie scripts.
2. **Few-Shot Prompting:** Use manual annotations to guide LLMs for automated annotation.
3. **Evaluation & Editing:** Review and correct errors in generated annotations.
4. **Parameter Tuning:** Perform an ablation study on decoding strategies and temperature to select optimal model parameters, using a novel benchmark for creative evaluation developed as part of this project, as no standard exists for creativity.
5. **Finetuning LLM:** Train a model to generate a JSON object containing details such as characters, script, background, mood (for music), expressions (for character sprites), and background edit prompts.
6. **Dynamic Visual Novel Pipeline:**
   (a) Input a rough story outline.
   (b) The finetuned model generates the story script in a fixed markdown format.
   (c) The **background edit prompt** optionally modifies the background (e.g., day $\rightarrow$ night).
   (d) JSON can be parsed by a rule based script to render the visual novel in real time using a browser based interface

4 Dataset Details

4.1 Data Collection

The dataset comprises 27 horror movie screenplays sourced from publicly available scripts at StudioBinder[4]. These screenplays represent diverse horror sub-genres including psychological

[4] https://www.studiobinder.com/blog/horror-movie-scripts/

horror, supernatural, slasher, and cosmic horror, providing varied narrative structures and atmospheric elements crucial for horror visual novel generation.

Category	Details
Format	Scripts downloaded in PDF format
Extraction Method	Extracted to plain text (.txt) using PDF parsing tools
Total Token Count	More than 650,000 tokens (est. 500,000-600,000 words)
Avg. Script Length	100-120 pages per screenplay
Total Scenes	Approx. 2,000-2,500 scenes

Table 1: Dataset Statistics

Initial Processing:

4.2 Annotation Schema

Our annotation format was designed to capture all essential elements for visual novel rendering:

Listing 1: Annotation Schema Structure

```
{
  "SCENE": {
    "LOCATION": "Highway",
    "MOOD": "Calm",
    "CHARACTERS": ["Narrator"],
    "BACKGROUND_IMAGE": "highway.png",
    "BACKGROUND_EDIT": "Daytime, empty
        dull road"
  },
  "SCRIPT": [{
    "CHARACTER": "Narrator",
    "LINE": "A dull highway. A crappy
        sedan roars by.",
    "EXPRESSION": "null"
  }],
  "PATHS": [{
    "CHOICE": "Continue driving",
    "TARGET": "crappy_car_interior",
    "STATE_CHANGE": null,
    "CONDITION": null
  }]
}
```

Schema Components:

- **SCENE**: Metadata including location, mood, characters present, and visual elements
- **SCRIPT**: Sequential dialogue and narration with character expressions
- **PATHS**: Branching choices enabling non-linear narrative exploration

4.3 Manual Annotation Subset

Six screenplays were fully manually annotated to serve as:

1. Few-shot examples for LLM-based annotation
2. Benchmark/reference data for quality evaluation
3. Ground truth for validation of automated annotations

This subset represents approximately 22% of the total dataset and provides high-quality training examples across different narrative styles.

5 Methodology

5.1 Data Preprocessing Pipeline

Our preprocessing pipeline consisted of four major phases:

5.1.1 Phase 1: Raw Text Extraction

Scripts were converted from PDF to plain text while preserving formatting cues (scene headings, character names in uppercase, stage directions in parentheses).

5.1.2 Phase 2: Intermediate Markdown Generation

Rather than directly generating JSON, we employed a two-stage approach.

Rationale: Reducing cognitive load on the model by separating content generation from structural formatting. This approach minimizes hallucinations and formatting errors (Kojima et al., 2022).

```
::SCENE::
LOCATION: Highway
MOOD: Calm
CHARACTERS: Narrator
BACKGROUND_IMAGE: highway.png
BACKGROUND_EDIT: "Daytime, empty dull
    road"

::SCRIPT::
- CHARACTER: Narrator
  LINE: A dull highway. A crappy sedan
      roars by.
  EXPRESSION: null

::PATHS::
- CHOICE: "Continue driving"
  TARGET: crappy_car_interior
  STATE_CHANGE: null
  CONDITION: null
```

5.1.3 Phase 3: Annotation Generation Experiments

We conducted systematic experiments with three prompting strategies to generate scene-level annotations.

Experiment 1: Zero-Shot Prompting (Ollama - Llama 3.2:1b) The model was prompted to annotate every scene, narration, dialogue, and stage direction without any examples. **Results:** Format adherence was good, but the overall content quality was poor, and PATH generation was missing. The model also introduced extraneous markdown syntax such as code blocks and bold markers.

Experiment 2: Few-Shot Prompting (Ollama - Llama 3.2:1b) This setup included manually annotated examples to guide the model. **Results:** Content quality improved significantly; however, format adherence degraded. The model often reverted to screenplay-style outputs instead of structured annotations.

Experiment 3: Few-Shot with Constrained Instructions (Gemini 2.5 Flash) Here, the prompt was enhanced with strict formatting rules, concrete examples, and explicit templates. Multiple few-shot examples from manual annotations were also provided. **Results:** Both format adherence and content quality were excellent, with complete PATH and EXPRESSION generation achieved. This configuration was therefore selected as the final approach.

5.1.4 Phase 4: Validation and Correction

We implemented an interactive validation tool to review and correct LLM-generated annotations.

Validation Tool Features:

- Parse and display structured annotations
- Scene-by-scene review interface
- Interactive editing of metadata, script lines, and paths
- Preservation of annotation structure during edits

Validation Statistics:

- Automated review of 21 LLM-generated scripts
- Average corrections per script: approximately 10% of total annotations
- Primary corrections: Character name consistency, expression inference, path generation completeness
- Manual validation time: Significantly reduced compared to full manual annotation

5.1.5 Phase 5: JSON Conversion

Final step employed a rule-based parser to convert validated Markdown to JSON.

Parser Features:

- Regex-based section splitting (::SCENE::, ::SCRIPT::, ::PATHS::)
- Field extraction with type validation
- Array construction for CHARACTERS and SCRIPT entries
- Null handling for optional fields

Quality Control:

- Manual review of 100% of JSON outputs
- Verification of structure compliance
- Cross-validation with original screenplay content

5.2 Fine-tuning Methodology

5.2.1 Model Selection

Base Model: Google Gemma 2-2B-IT (Instruction-Tuned)

- Source: https://huggingface.co/google/gemma-2-2b-it
- Parameters: 2 billion (manageable for GPU memory constraints)
- Architecture: Transformer decoder with RoPE positional embeddings
- Pre-training: Instruction-following and conversational tasks

Rationale: Gemma models demonstrate strong performance on creative writing tasks while being small enough for efficient fine-tuning on consumer hardware (T4/A100 GPUs available on Kaggle).

5.2.2 Training Data Preparation

Data Augmentation Strategy: We generated three instruction-response formats from each scene to maximize training diversity:

Task	Complete Scene Generation
Input	Location, mood, character list
Output	Full narration and dialogue sequence

Table 2: Complete Scene Generation task description.

```
<start_of_turn>user
{instruction}<end_of_turn>
<start_of_turn>model
{response}<end_of_turn>
```

Task	Dialogue Continuation
Input	First half of scene script
Output	Continuation of dialogue

Table 3: Dialogue Continuation task description.

Task	Emotion-Conditioned Generation
Input	Character, emotion, location
Output	Contextually appropriate dialogue line

Table 4: Emotion-Conditioned Generation task description.

Gemma Chat Template

Statistic	Value
Total scenes	∼5,891
Number of scripts	27
Total tokens	600,000+
Average input length	50–100 tokens
Average output length	100–300 tokens

Table 5: Dataset statistics overview.

Dataset Statistics:

5.2.3 Memory-Efficient Training Configuration

Listing 2: Quantization Configuration

```
bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.
        bfloat16,
    bnb_4bit_use_double_quant=True
)
```

Memory Reduction: Approximately 75% reduction in model memory footprint (from 8GB to 2GB).

Listing 3: LoRA Configuration

```
lora_config = LoraConfig(
    r=8,
    lora_alpha=16,
    target_modules=["q_proj", "k_proj",
                    "v_proj", "o_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)
```

Trainable Parameters: Approximately 0.5% of total parameters (reducing from 2B to approximately 10M trainable parameters).

5.2.4 Training Setup

Key hyperparameters: 3 epochs; batch size 1 (accum. 16 $\Rightarrow$ eff. 16); LR 2×10^{-4} with 0.1 warmup and cosine decay; optimizer = Paged AdamW (8-bit), weight decay 0.001, max grad norm 0.3.
Infrastructure: Kaggle T4 (16GB VRAM), ∼16h for 3 epochs, checkpoints every 100 steps (latest only).
Memory optimizations: gradient checkpointing; 8-bit optimizers; manual cache clearing; `use_cache=False`; single checkpoint retention.
Artifacts: LoRA weights (`adapter_model.safetensors`), config (`adapter_config.json`), tokenizer, and `training_config.json`.

6 Benchmarking

6.1 Introduction

This section introduces a tournament-based evaluation framework for ranking creative text generation outputs, addressing the limitations of traditional 'naive' AI scoring methods that suffer from score compression and insufficient discriminative power. We implement knockout elimination, Swiss tournament and ELO based systems, validating our approach through extensive testing.

6.2 Core Axioms

1. **Relative Quality Assessment**: Creativity has no absolute score—quality is measured relatively, like internal energy in thermodynamics, being inherently contextual and comparative.
2. **Transitivity of Creative Comparison:** If $A > B$ and $B > C$, then $A > C$, enabling complete orderings from pairwise comparisons (though empirical limitations exist).
3. **LLM Comparison Superiority**: LLMs perform better at pairwise text comparison than absolute scoring (which suffers from rating inflation and score compression), motivating our tournament-based approach.

6.3 Tournament Methodology Framework

6.3.1 Algorithm 1: Naive Independent Scoring

Baseline comparison method

- Score each text individually based on a detailed prompt (0-100 scale)

- Rank by absolute scores
- Demonstrates typical scoring limitations: compression and poor discrimination

6.3.2 Algorithm 2: Knockout Tournament

Single-elimination with complete ranking

- **Setup**: N texts paired randomly; byes assigned if N not power of 2
- **Main Tournament**: Pairwise comparisons eliminate losers until champion determined
- **Recursive Ranking**: For rank k, run tournament among texts defeated by top (k-1) ranked texts
- **Output**: Complete 1-to-N ranking

6.3.3 Algorithm 3: Swiss Tournament

Non-elimination with performance-based pairing

- **Setup**: N texts, R = $\lceil \log_2(N) \rceil$ rounds
- **Round 1**: Random pairing, winners get 1 point
- **Rounds 2-R**: Pair texts with similar scores, avoid rematches
- **Final Ranking**: Sort by points; break ties using Buchholz score (sum of opponents' scores)
- **Output**: Complete ranking with strength-of-schedule consideration

6.3.4 Algorithm 4: ELO Rating System

Continuous rating with dynamic pairing

- **Setup**: N texts start with rating 1500, adaptive K-factors (32→24→16 based on experience)
- **Pairing Strategy**: Swiss-style pairing based on ELO ratings, avoid rematches
- **Rating Updates**: Winner gains rating, loser loses rating based on expected outcome
- **Final Ranking**: Sort by ELO rating; break ties using Buchholz score (sum of opponents' ratings)
- **Output**: Complete ranking with skill-based ratings and opponent strength consideration

6.4 Validation

Validation used 16 text files (file_01.txt to file_16.txt) with incrementally increasing creativity levels, enabling sensitivity assessment to known quality gradients across all four ranking methodologies.

6.4.1 Example Tournament Comparison

The following demonstrates how individual pairwise comparisons are conducted during tournament execution (see Appendix for detailed output).

This exemplifies the detailed reasoning provided by the LLM judge for each comparison, enabling transparency and verification of the ranking process.

6.4.2 Observations and Conclusions

- **ELO Rating System**: Superior methodology with continuous skill assessment and dynamic ratings. Correctly identified file_14.txt as champion (1602 ELO, 7-0 record) and file_01.txt as lowest (1398 ELO, 0-7 record). Precise differentiation across all tiers with rating spread of 204 points and varied win rates.
- **Swiss Tournament**: Strong discriminative power with file_16.txt achieving perfect 6-point record and file_01.txt scoring zero. However, shows clustering limitations with multiple 4-point and 3-point ties, unable to clearly distinguish file_14.txt as the top performer.
- **Knockout Tournament**: Reliable extreme identification (file_16.txt first, file_01.txt last) but significant mid-range variance. File_14.txt ranked only second despite being the true best performer, demonstrating single-elimination limitations.
- **Naive Scoring**: Catastrophic failure with 87.5% score compression (14/16 files at 85/100). Cannot differentiate between file_14.txt (ELO champion) and mid-tier performers, confirming absolute scoring inadequacy.

Overall Assessment: $ELO > Swiss > Knockout > Naive$ hierarchy established, with ELO providing optimal accuracy for creative text evaluation and motivating adoption for Gemma model optimization studies.

7 Ablation Study: Decoding Strategy Performance Analysis

7.1 Introduction

This ablation study examines the performance of various text generation decoding strategies across two distinct linguistic tasks: creative text generation and syntax adherence. We evaluate five primary decoding approaches—sampling, top-k, top-p, beam search, and greedy decoding—across multiple hyperparameter configurations using an enhanced ELO tournament system with Buchholz scoring for robust tie-breaking. We conducted comprehensive tournaments using 74 distinct configurations per task, the description of which are as following:

1. **Sampling:** Temperature-controlled stochastic selection (T $\in$ [0.1, 1.0])
2. **Top-K:** Selection from top-k most probable tokens (k $\in \{10, 30, 50\}$, T $\in$ [0.1, 1.0])
3. **Top-P (Nucleus):** Cumulative probability threshold sampling (p $\in \{0.5, 0.9, 0.98\}$, T $\in$ [0.1, 1.0])
4. **Beam Search:** Breadth-first exploration (beam width $\in \{3, 5, 10\}$)
5. **Greedy Decoding:** Deterministic maximum probability selection

7.2 Results and Analysis

7.2.1 Creative Text Generation Task

Overall Performance Rankings The creative text generation task revealed distinct performance patterns across decoding strategies. **Top-K sampling with T=0.7 and k=30** achieved the highest performance (ELO: 1646), demonstrating superior balance between diversity and coherence for creative tasks.

See Appendix Image 1 for comprehensive ELO ranking showing all 74 configurations ranked by performance.

Strategy Performance Hierarchy:

1. **Greedy:** 1564.0 (highest average, single configuration)
2. **Beam Search:** 1541.7 (range: 1468-1593)
3. **Sampling:** 1522.4 (range: 1436-1596)
4. **Top-K:** 1499.5 (range: 1403-1646)
5. **Top-P:** 1486.7 (range: 1353-1584)

Temperature Sensitivity Analysis Temperature sensitivity varied significantly across strategies:

Sampling Strategy: Exhibited clear optimal temperature at T=0.5 (ELO: 1596), with performance degradation at both extremes. Lower temperatures (T=0.1-0.3) produced overly conservative outputs, while higher temperatures (T¿0.8) introduced excessive randomness.

Top-K Strategy: Demonstrated peak performance at moderate-to-high temperatures (T=0.9, avg ELO: 1543), suggesting that creative tasks benefit from increased stochasticity when constrained by top-k selection.

Top-P Strategy: Showed relatively stable performance across temperatures, with slight preference for moderate values (T=0.6, avg ELO: 1510.7).

Parameter-Specific Analysis Top-K Parameter Optimization:

- k=30 emerged as optimal (avg ELO: 1514.6)
- k=10 showed reduced performance (avg ELO: 1490.3), likely due to insufficient diversity
- k=50 demonstrated moderate performance (avg ELO: 1493.7), suggesting diminishing returns

Top-P Parameter Optimization:

- p=0.9 achieved best performance (avg ELO: 1508.4)
- p=0.98 showed comparable results (avg ELO: 1485.1)
- p=0.5 significantly underperformed (avg ELO: 1466.6), indicating excessive restriction

Beam Search Width Analysis:

- Optimal beam width: 5 (ELO: 1593)
- Beam width 3: Good performance (ELO: 1564)
- Beam width 10: Significant degradation (ELO: 1468), suggesting over-exploration penalties

7.2.2 Syntax Adherence Task

Overall Performance Rankings The syntax adherence task showed markedly different optimal configurations, with **Top-P sampling (T=0.9, p=0.98)** achieving peak performance (ELO: 1622).

See Appendix Image 2 for comprehensive ELO ranking showing all 74 configurations ranked by performance.

Strategy Performance Hierarchy:

1. **Sampling:** 1519.2 (range: 1500-1564)
2. **Top-K:** 1509.1 (range: 1436-1615)
3. **Top-P:** 1488.9 (range: 1347-1622)
4. **Beam Search:** 1478.0 (range: 1465-1501)
5. **Greedy Decoding:** 1436.0 (single configuration)

Task-Specific Strategy Adaptations Sampling Dominance: Unlike creative tasks, sampling achieved the highest average performance, with optimal temperature at T=0.7 (ELO: 1564). This suggests that syntax tasks benefit from controlled randomness without excessive constraints.

Top-K Temperature Preferences: Syntax tasks showed preference for higher temperatures (T=0.7-0.9), contrasting with creative tasks that favored more moderate values.

Greedy Degradation: Greedy decoding performed significantly worse on syntax tasks (ELO: 1436) compared to creative tasks (ELO: 1564), indicating that deterministic selection impairs syntactic flexibility.

Cross-Task Parameter Generalization Temperature Shift: Syntax tasks consistently preferred higher temperatures across all strategies, suggesting greater tolerance for stochasticity when structural constraints are primary.

Beam Search Limitations: Beam search showed reduced effectiveness for syntax tasks, with optimal width increasing to 10 (ELO: 1501) compared to creative tasks' preference for width 5.

7.3 Discussion

7.3.1 Task-Dependent Strategy Selection

Our findings reveal fundamental differences in optimal decoding strategies between creative and syntactic tasks:

Creative Tasks favor structured exploration with moderate constraints (Top-K with k=30, T=0.7), balancing novelty with coherence. The superior performance of beam search variants suggests that systematic exploration benefits creative generation.

Syntax Tasks require greater stochastic flexibility, with unconstrained sampling and high-p nucleus sampling performing best. This indicates that syntactic adherence benefits from broader probability distributions during token selection.

7.3.2 Temperature as a Universal Tuning Parameter

Temperature emerged as the most critical hyperparameter across all strategies:

- **Creative tasks:** Optimal range T=0.5-0.7
- **Syntax tasks:** Optimal range T=0.7-0.9

This systematic shift suggests that task complexity and constraint types fundamentally alter the optimal exploration-exploitation balance.

7.3.3 Constraint Mechanism Effectiveness

Different constraint mechanisms showed varying effectiveness:

Top-K constraints proved most effective for creative tasks, providing structured diversity while maintaining quality. The optimal k=30 suggests a sweet spot between diversity and focus.

Top-P constraints excelled in syntax tasks, particularly at high thresholds (p=0.98), indicating that probability-based truncation better preserves syntactic validity than fixed vocabulary limits.

Beam search showed task-dependent effectiveness, excelling in creative contexts but struggling with syntactic constraints, possibly due to its deterministic exploration pattern.

7.3.4 Buchholz Scoring Insights

The Buchholz scoring system revealed subtle performance differences obscured by simple win rates. Configurations with identical ELO ratings often showed significant Buchholz score variations, indicating different competitive contexts. This highlights the importance of considering opponent strength in comparative evaluations.

7.3.5 Visual Analysis Insights

The comprehensive visualization suite provides detailed insights into strategy performance patterns, parameter interactions, and cross-task differences that supplement the numerical analysis.

Strategy-Specific Performance Matrices See Appendix Images 3 and 4 for Top-K and Top-P strategy performance matrices comparing creative and syntax tasks.

Matrix Analysis Insights: The strategy-specific matrices reveal fundamental differences in parameter sensitivity between tasks. Creative tasks show concentrated high-performance regions with sharp boundaries, indicating specific optimal parameter combinations (notably Top-K k=30 at T=0.7). Syntax tasks display more distributed performance with smoother transitions across parameters, suggesting greater tolerance for parameter variation. The color gradients clearly demonstrate that creative tasks require precise parameter tuning, while syntax tasks benefit from broader exploration ranges.

Temperature Sensitivity Analysis See Appendix Image 5 for temperature sensitivity curves comparing creative and syntax tasks.

Temperature Sensitivity Insights: The temperature curves reveal striking task-dependent patterns. Creative tasks exhibit a characteristic inverted-U curve with a sharp peak at T=0.5 (ELO: 1596), demonstrating sensitivity to both under-exploration (low T) and over-exploration (high T). Syntax tasks show optimal performance at T=0.7 (ELO: 1564) with a broader plateau, indicating greater tolerance for stochastic variation. This systematic temperature shift of 0.2 units higher for syntax tasks suggests that structural constraints require more exploration flexibility than creative coherence.

Parameter Variation Analysis for Optimal Temperature Range (T=0.7-0.9) This section analyzes how parameter choices affect performance within the optimal temperature range identified across both tasks.

See Appendix Images 6 and 7 for detailed parameter variation analysis across temperatures T=0.7, 0.8, and 0.9.

Parameter Variation Analysis Insights: The parameter variation analysis across the optimal temperature range (T=0.7-0.9) reveals critical insights into task-specific optimization patterns. For Top-K strategy, creative tasks show consistent k=30 optimality across all temperatures, but with declining absolute performance as temperature increases from 0.7 to 0.9. Syntax tasks demonstrate greater parameter tolerance and maintain stronger performance at higher temperatures, particularly at T=0.9.

For Top-P strategy, the analysis confirms that higher p-values (0.9-0.98) consistently outperform lower values across all temperatures. Syntax tasks particularly benefit from p=0.98 at T=0.9, achieving peak performance (ELO: 1622), while creative tasks show optimal Top-P performance at more moderate temperatures (T=0.6-0.7). This systematic analysis validates that temperature emerges as the most critical hyperparameter, with task-specific optimal ranges requiring different parameter complementarity patterns.

Key Insights from Parameter Variation Analysis:

Temperature Sensitivity Patterns:

- **Creative Tasks:** Show declining performance as temperature increases from 0.7 to 0.9, with optimal performance consistently at T=0.7
- **Syntax Tasks:** Maintain or improve performance at higher temperatures, with peak effectiveness at T=0.9

Parameter Interaction Effects:

- **Top-K Strategy:** k=30 remains optimal across temperature ranges for creative tasks, while syntax tasks show greater parameter tolerance
- **Top-P Strategy:** Higher p-values (0.9-0.98) consistently outperform lower values, with syntax tasks particularly benefiting from p=0.98 at T=0.9

Cross-Task Parameter Transferability:

- **Limited Transferability:** Optimal parameters identified for one task type do not generalize effectively to the other
- **Temperature Dominance:** Temperature emerges as the most critical hyperparameter, with task-specific optimal ranges requiring different parameter complementarity

7.4 Practical Implications

7.4.1 Strategy Selection Guidelines

Based on our findings, we recommend:

For Creative Text Generation:

- **Primary choice:** Top-K sampling (k=30, T=0.7)

- **Alternative:** Beam search (width=5)
- **Avoid:** Low-temperature configurations ($T < 0.5$)

For Syntax Adherence:

- **Primary choice:** Top-P sampling (p=0.98, T=0.9)
- **Alternative:** Temperature sampling (T=0.7)
- **Avoid:** Greedy decoding or low-temperature configurations

Compromise configurations: Top-K with k=30, T=0.8 provides reasonable performance across both tasks.

8 Evaluation Options

8.1 Evaluation of Benchmark

A benchmark to evaluate the creativity of generated text has been developed as part of this project and has demonstrated initial effectiveness in assessing creativity and coherence. To verify its robustness, it will be compared against **human-evaluated rankings**, which will serve as a gold standard for evaluating the benchmark's efficacy.

8.2 Evaluation of Text

Text generation will be evaluated using multiple approaches:

- **Benchmark (updated with Elo Rating / Ensemble of Judges):** Applied to generated text using comparative rankings and aggregated scores from multiple evaluators. Detailed methodology is explained in the benchmark subpart of the progress section.
- **Self-BLEU:** Multiple responses are generated for a single prompt, and the average BLEU score is calculated by comparing each response against the others. A **low Self-BLEU score** indicates high diversity, suggesting more creative outputs.
- **Human Evaluation:** Human raters assess creativity, coherence, and overall quality to validate the automated evaluation results.

Acknowledgements

This report is made by Amisha, Aaryan Kashyap and Jatin Agrawal with the great honor of submitting it to Professor Manish Shrivastava. We want to thank both the professor Manish Shrivastava and the TA, Miss. Ketaki Shetye who provided us with guidance whenever required. The making of this project was a really good experience for us. It was a task full of mirth for us.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36.

Gemma Team. 2024. Gemma: Open models based on Gemini technology. Technical report, Google DeepMind.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213.

Hairong Liu, Mingda Zhang, and Shih-Fu Chang. 2020. ScriptBase: A database for understanding movies. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 16(2):1–23.

Mark O. Riedl and Vadim Bulitko. 2013. Interactive narrative: An intelligent systems approach. *AI Magazine*, 34(1):67–77.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Marilyn A. Walker, Grace I. Lin, and Jennifer Sawyer. 2011. Towards automatic generation of a planning-based storyboard for videos. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 7, pages 58–64.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting

elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, volume 27.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

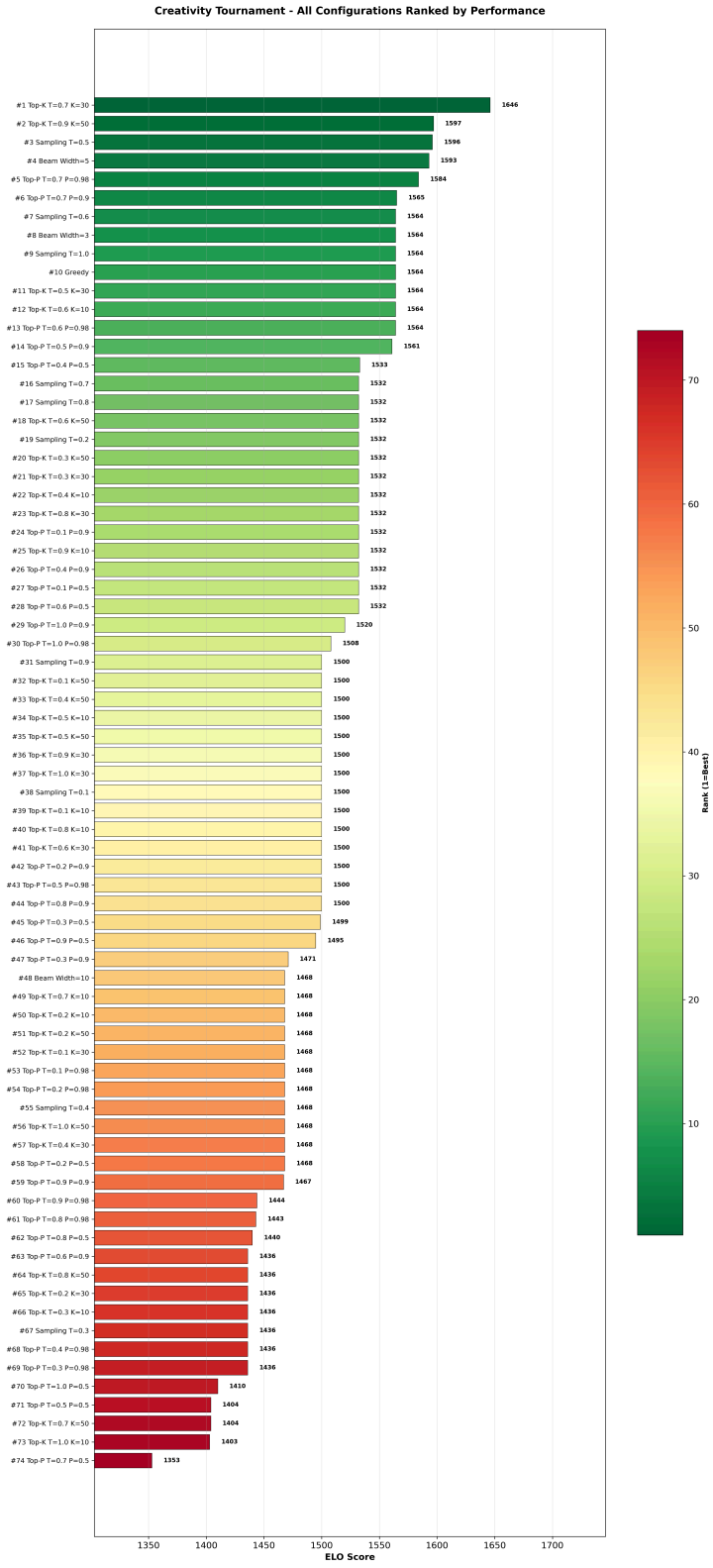
Appendix: Ablation Study Figures

This appendix contains comprehensive visualizations supporting the ablation study analysis presented in the main text.

Comprehensive ELO Rankings

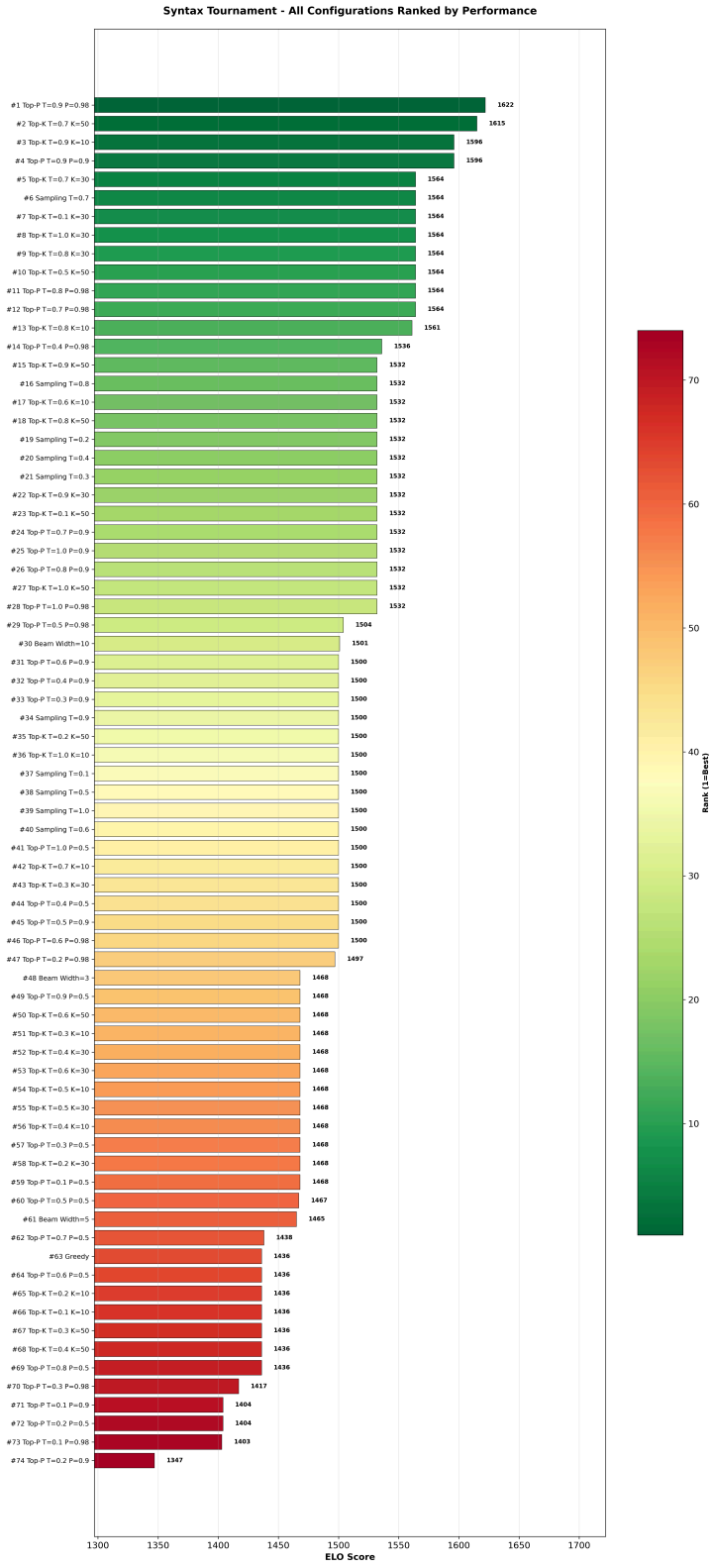
Appendix Image 1: Creative Text Generation - Comprehensive ELO Ranking

Complete ELO ranking of all 74 decoding configurations for the creative text generation task, ordered by performance.



Appendix Image 2: Syntax Adherence - Comprehensive ELO Ranking

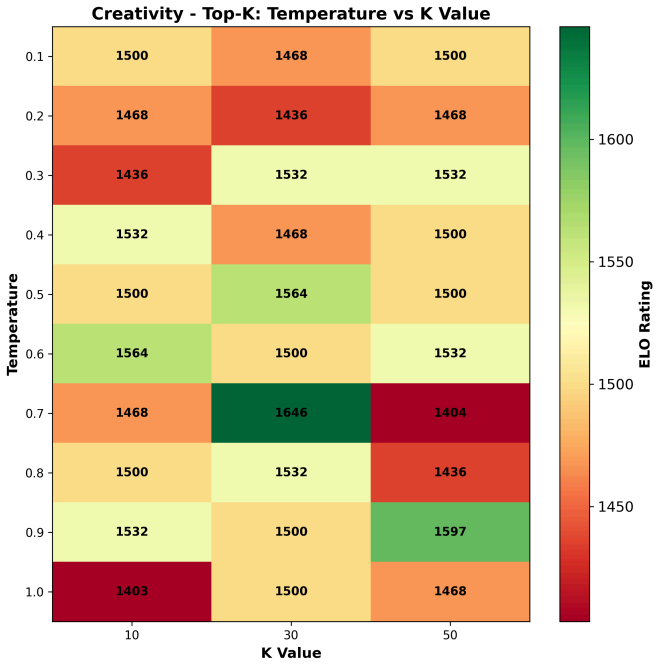
Complete ELO ranking of all 74 decoding configurations for the syntax adherence task, ordered by performance.

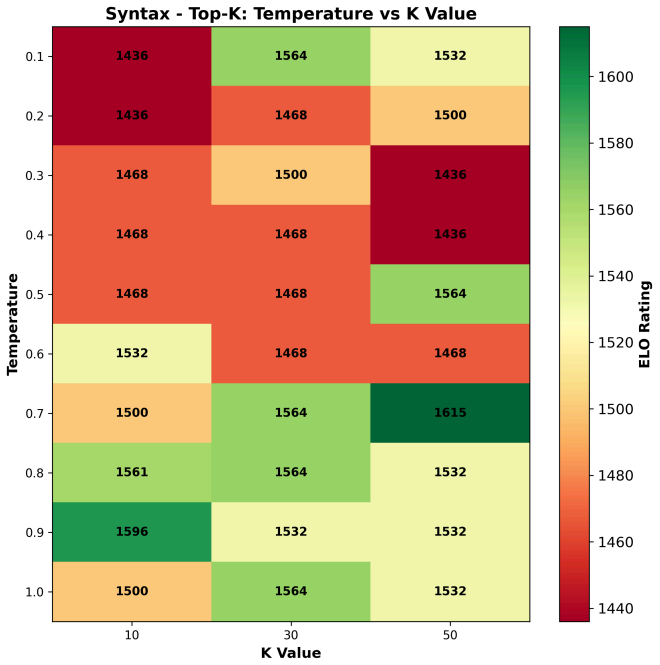


Strategy-Specific Performance Matrices

Appendix Image 3: Top-K Strategy Performance Matrices

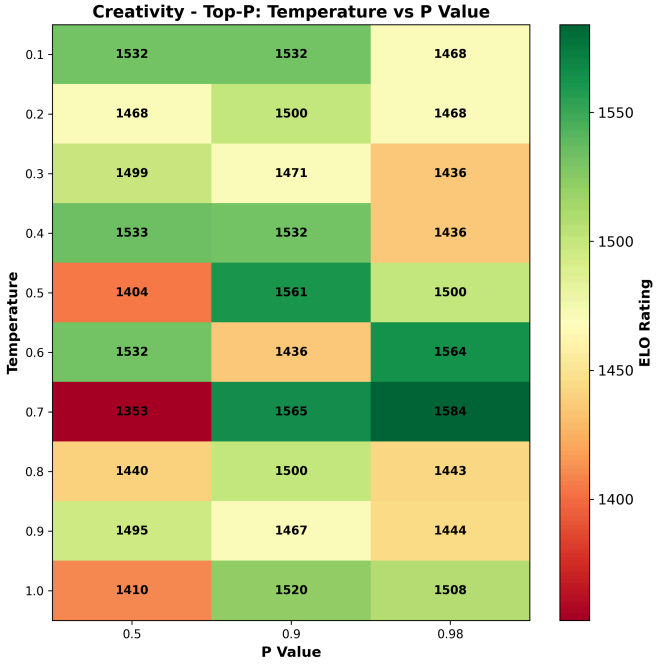
Heat map matrices comparing Top-K strategy performance across both tasks.

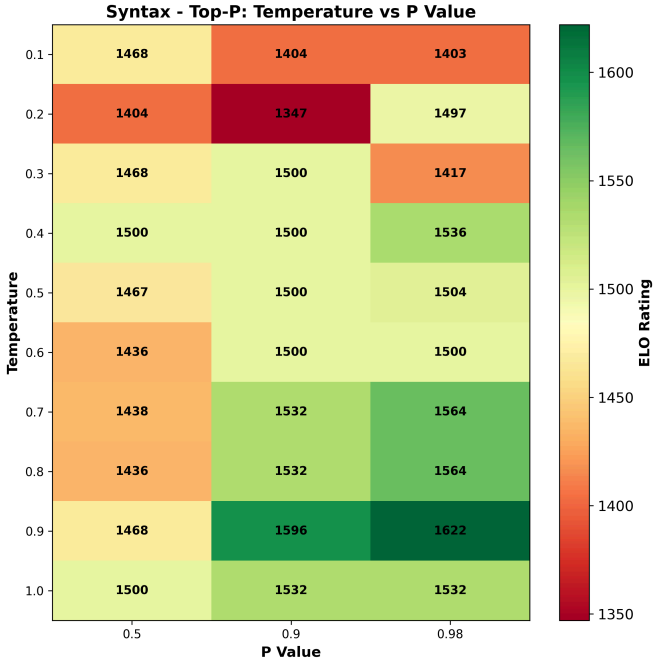




Appendix Image 4: Top-P Strategy Performance Matrices

Heat map matrices comparing Top-P (nucleus sampling) strategy performance across both tasks.

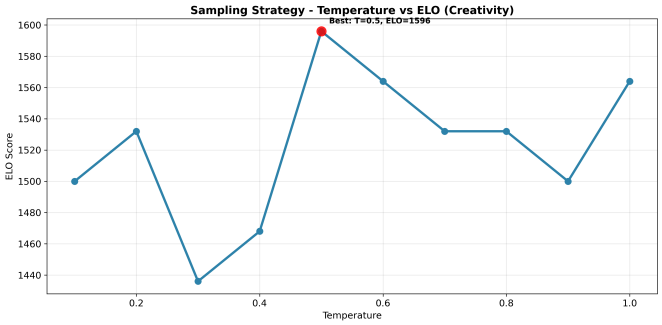


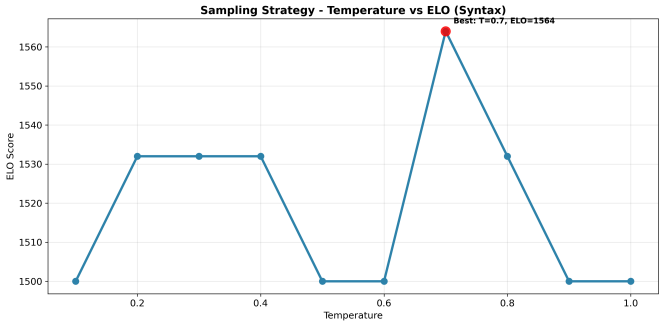


Temperature Sensitivity Analysis

Appendix Image 5: Temperature Sensitivity Curves

Temperature sensitivity curves for pure sampling strategy across both tasks.

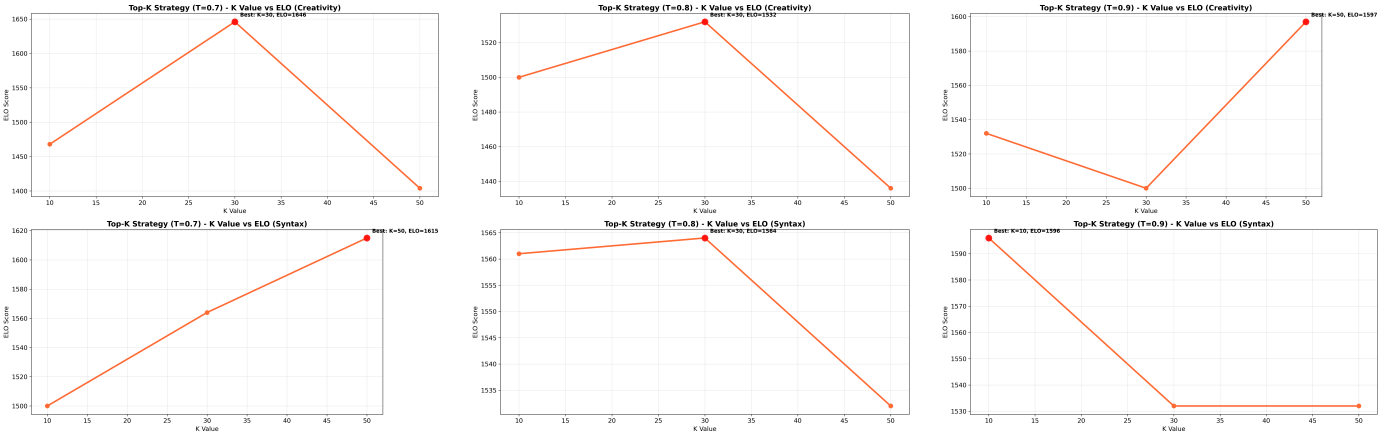




Parameter Variation Analysis

Appendix Image 6: Top-K Parameter Variation Analysis (T=0.7-0.9)

Top-K parameter (k-value) performance across the optimal temperature range for both tasks.



Appendix Image 7: Top-P Parameter Variation Analysis (T=0.7-0.9)

Top-P parameter (p-value) performance across the optimal temperature range for both tasks.