

CONSTRAINT-AUGMENTED NEURAL NETWORKS FOR SHAKESPEAREAN VERSE: INTEGRATING PROSODIC FIDELITY WITH TRANSFORMER MODELS

MEHULKUMAR H KANTARIA¹, SAHANA EDWIN², VISHAL NAMIREDDY³
S. KRISHNAKUMARI⁴, JAKKAPU NAGALAKSHMIDEVI⁵, DR GANTA JACOB VICTOR⁶

¹Independent Researcher, Sadhu Vasvani Road, Rajkot, India.

²Professor, Department of Computer Science & IT, Garden City University, Bengaluru, India.

³Full Stack Developer (Java, Cloud, DevOps, Front-End Engineering), Slesha IT Inc, Dallas, TX,USA.

⁴Department of Electronics and Communication, Loyola ICAM college of Engineering and Technology, Chennai, India.

⁵Assistant Professor, Department of computer science and engineering, Aditya University, Surampalem, India.

⁶Associate Professor, Department of Computer Science and Engineering, Koneru Lakhmaiah Education Foundation, Vaddeswaram, Guntur,India.

¹kantariamehul@gmail.com, ²sahanaedwin78@gmail.com, ³vishaljv3@gmail.com,

⁴krishnakumari.s@licet.ac.in, ⁵nagajakkapu@gmail.com, ⁶jacob.victor@kluniversity.in

ABSTRACT

This research proposes a methodology for augmenting prosodic constraints to enhance the creation of Shakespearean poetry in neural models of language. Contemporary transformer-based generators are proficient in fluency but often do not satisfy the stringent metrical and rhyming standards of formal poetry. To fill this gap, the proposed technique uses specialized prosody embeddings as well as prosody-aware attention mechanisms to combine explicit prosodic information such syllable counts, stress patterns, along with rhyme suffixes. The model is guided toward simultaneous optimization of linguistic quality as well as prosodic integrity by a composite training goal that combines cross-entropy loss with distinct meter and rhyme penalties and optional reinforcement learning. Experiments on a prosodically annotated Shakespearean corpus demonstrate substantial gains in meter accuracy, rhyme correctness, and overall stylistic adherence compared with standard GPT-based baselines, without sacrificing coherence or fluency. The results highlight the effectiveness of embedding phonological structure directly into neural architectures and offer a scalable approach for controllable poetic and stylistically constrained text generation.

Keywords: *Prosodic Constraint Augmentation, Shakespearean Verse Generation, Neural Language Models, Meter and Rhyme Modelling, Controllable Text Generation.*

1. INTRODUCTION:

Formal verse especially Shakespearean iambic pentameter and canonical rhyme patterns has long been a benchmark for computational prosody and stylistic generation. Early statistical and RNN-based systems captured local patterns but frequently failed to satisfy meter and rhyme [1] [2]. With large pretrained transformers, fluency and lexical richness improved, yet adherence to formal prosodic structure remained elusive [3] [4]. This paper originates at the intersection of computational prosody, neural controllable generation, and Shakespearean style modelling. Shakespearean prosody is governed principally by meter (regular stress-unstress

patterns; iambic pentameter predominates) and rhyme (phonological suffix similarity at line endings). Computational tools pronunciation lexica (CMUdict), G2P models, stress detectors, and syllable-segmentation algorithms make token-level prosodic annotation possible [5] [6]. Prior controllable-generation work (masking, prefix control, RL steering) provides mechanisms to bias outputs but rarely integrates fine-grained phonological features directly into the core attention/decoding computations of modern autoregressive models [7] [8]. The present work draws on these linguistic resources and controllable-generation paradigms to close that gap [9] [10]. Neural generators must satisfy two competing

demands when producing Shakespearean verse: (1) maintain high language-model fluency and semantic coherence, and (2) obey strict phonological constraints (meter and rhyme) that are defined at the syllable/phoneme level rather than at the token level [11] [12]. Standard Transformer decoders lack explicit phonological awareness and therefore tend to violate meter and rhyme unless heavily post-processed or constrained [13] [14]. The core research problem is: How can we inject token-level prosodic knowledge into a transformer so the model jointly optimizes for fluency and prosodic fidelity without catastrophic semantic drift? Solving this problem advances computational creativity and controllable text generation by demonstrating a principled way to incorporate phonological structure into deep generative models [15] [16].

From a practical standpoint, it allows for automated generation that pays more heed to poetic forms; this is applicable to creative-assistive tools, education, and digital editions. From a methodological one, it shows how constraints driven by structure and language can be encoded at the levels of model architecture and loss function, instead of being applied as external filters. More general studies on genre and language style transfer and limited generation may benefit from the suggested approach as well. There has been a rise in both interest in creative-AI and criticisms over its controllability and style integrity. Shakespearean verse and other culturally and historically sensitive genres run the danger of becoming metrically nonsensical or archaic due to naïve generation. The modern need for explainable and controlled generation is fulfilled by explicit prosodic augmentation, which improves transparency (meter/rhyme can be measured), allows for tuneable form-meaning trade-offs using loss weights, and enables human-in-the-loop adjustment for high-value products. Prosody embeddings, assessment metrics that take prosody into account, and the datasets used to conduct this robust G2P for Early Modern orthography are all excellent resources in and of themselves. The principal objective is to design, implement, and empirically validate a Prosodic Constraint Augmentation framework that: (1) extracts per-token prosodic features (stress, syllable counts, rhyme suffixes), (2) encodes them into prosody embeddings injected into attention and decoder computations, and (3) trains the network with a composite objective (cross-entropy + differentiable meter/rhyme penalties, with optional RL fine-tuning) so that generated verse approaches canonical Shakespearean prosody while retaining high language quality. The paper operationalizes this objective through a detailed architecture, dataset

construction, rigorous baselines, automatic metrics, ablations, and human evaluation.

2. BACKGROUND & RELATED WORK:

The background for this work situates prosodic augmentation at the intersection of neural poetry generation, computational prosody, and constraint-based controllable generation. Early neural poetry systems relied on LSTM/RNN architectures that could capture local sequence patterns but tended to produce weak or inconsistent meter and rhyme; more recent GPT-style autoregressive models improve fluency and lexical variety but still struggle to obey formal prosodic patterns without explicit supervision [17] [18]. Research in prosody for NLP has produced robust tools for stress detection, meter tagging, and phoneme-based rhyme similarity (CMUdict/G2P pipelines and learned stress predictors), which enable automatic scoring and supervision of metrical/rhyming structure. Parallel lines of work on controllable generation masking, prefix control, soft vs. hard constraint losses, and RL-based steering offer mechanisms to impose structure but rarely target phonological constraints specifically. In the narrow subfield of Shakespearean style transfer, iambic pentameter and canonical rhyme schemes are well understood linguistically yet remain challenging for unconstrained neural models [19] [20]. Crucially, there is a clear gap: few systems tightly integrate explicit, token-level prosodic features with deep generative architectures to jointly optimize fluency and formal prosody an issue this paper addresses.

3. SHAKESPEAREAN PROSODY: LINGUISTIC & COMPUTATIONAL FOUNDATIONS:

Shakespearean verse is governed by two tightly interwoven prosodic phenomena: meter (regular patterns of stressed and unstressed syllables) and rhyme (phonological suffix similarity at line endings). Modelling these formally is essential for any constraint-augmented generative system.

3.1. Iambic Pentameter Structure:

Iambic pentameter is the dominant metrical frame in Shakespeare's sonnets and many of his dramatic lines. Each canonical line consists of five metrical feet, each foot an iamb (unstressed → stressed). If we encode an unstressed syllable as 0 and a stressed syllable as 1, the ideal target pattern for a complete iambic pentameter line of length $L=10$ is

$$T = (0,1,0,1,0,1,0,1,0,1).$$

In practice Shakespeare often allows controlled deviations: substitutions (trochee at the line-initial foot), feminine endings (an extra unstressed syllable at the line end), elisions, and occasional catalexis (missing final unstressed syllable). To represent an observed line i we extract a stress sequence

$$S^{(i)} = (s_1^{(i)}, s_2^{(i)}, \dots, s_{L_i}^{(i)}, s_k^{(i)} \in \{0,1\},$$

where L_i is the syllable count for the line (which may differ from 10). A per-line meter match score can be defined using a normalized Hamming-match against a suitably aligned target (allowing a one-syllable shift to accommodate feminine endings):

$$m_i = \max_{\delta \in \{-1,0,1\}} \frac{1}{L_T} \sum_{k=1}^{L_T} 1(s_{k+\delta}^{(i)} = T_k),$$

where L_T is the length of the canonical target (usually 10) and out-of-range indices are ignored (or treated as mismatches). The max over small shifts δ allows forgiving alignment for common metrical variants.

3.2. Rhyme schemes:

Shakespeare's sonnets follow the well-known quatrain + couplet pattern (ABAB CDCD EFEF GG), but dramatic verse uses more flexible rhyming. Rhyme is fundamentally phonological: two line-end tokens rhyme when their stressed vowel and subsequent phoneme sequence match to a high degree. Using phonemic transcriptions (e.g., from CMUdict), denote the phoneme sequence of the final stressed vowel and its following coda for line i as $P^{(i)} = (p_1^{(i)}, \dots, p_{r_i}^{(i)})$. A simple rhyme similarity is the normalized longest common suffix length.

$$R(i, j) = \frac{\ell_{\text{suffix}}(P^{(i)}, P^{(j)})}{\max(r_i, r_j)} \in [0,1],$$

where ℓ_{suffix} returns the length of the longest identical suffix. A binary rhyme indicator for a target rhyme class (e.g. expecting A vs B) is $1(R(i, j) > \tau)$ for threshold τ (typical $\tau \in [0.6, 0.9]$ depending on strictness).

3.3. Computational Prosody Tools and Scoring:

Practically, prosodic modelling leverages resources and algorithms that map orthography to stress and syllable structure:

Pronunciation lexica (CMU Pronouncing Dictionary): provides canonical phoneme sequences and primary/secondary stress markers for many English words; used to extract $S^{(i)}$ and $P^{(i)}$ for in-vocabulary tokens. Stress annotation & grapheme - phoneme models: out-of-vocabulary words (archaic spellings, names) are handled by G2P models or neural stress predictors that estimate per-syllable

stress labels. Syllable segmentation: approximate syllable counts via vowel nucleus detection in phoneme sequences; used to compute L_i and detect feminine endings or elisions. Prosody scoring: combine meter, syllable and rhyme measures into a single soft score usable in training. For a corpus of N lines we can define the empirical meter consistency

$$P_{\text{meter}} = \frac{1}{N} \sum_{i=1}^N 1(m_i \geq \eta),$$

where η is a threshold (e.g. 0.8) for an acceptable per-line meter match. A continuous alternative averages the per-line match:

$$\bar{P}_{\text{meter}} = \frac{1}{N} \sum_{i=1}^N m_i.$$

To capture multiple prosodic targets, a composite prosody score for a generated stanza (or batch) is convenient:

ProsodyScore = $w_m \bar{P}_{\text{meter}} + w_r \bar{R} - w_s \bar{\Delta}_{\text{syll}}$, where \bar{R} is the average expected rhyme similarity across paired lines, $\bar{\Delta}_{\text{syll}}$ is the mean absolute syllable-count deviation from the canonical target (penalizing extra/missing syllables), and $w_m, w_r, w_s \geq 0$ are tunable weights. This soft objective is differentiable when m_i and R are implemented via smooth approximations (e.g., soft alignment, phoneme-embedding similarity), enabling incorporation into neural training as a prosody loss term:

$$\mathcal{L}_{\text{prosody}} = -\text{ProsodyScore}.$$

Taken together, these linguistic descriptions and computational tools provide a principled bridge between metrical theory and implementable constraints: pronunciation lexica and G2P supply the basic phonological atoms, alignment-aware match functions quantify meter fidelity including typical Shakespearean deviations, and suffix-based rhyme metrics operationalize rhyming for both evaluation and loss formation.

4. PROPOSED METHOD - PROSODIC CONSTRAINT AUGMENTATION:

4.1. Model overview:

Researcher build on a Transformer / GPT-style autoregressive backbone and augment it with a lightweight prosody encode-decode pipeline that produces per-token prosodic features and injects them into the Transformer's attention and output layers. At generation time the model receives a (semantic) prompt and optionally a target prosodic template (e.g., iambic pentameter + rhyme scheme). The pipeline (1) maps tokens \rightarrow

phoneme/stress/syllable features via a G2P + stress tagger, (2) encodes those prosodic features into dense vectors e_{prosody} , (3) injects e_{prosody} into attention (prosody-aware attention) and into the decoder head for final token scoring, and (4) evaluates prosody via a differentiable prosody loss used during training. This keeps the strong language modelling abilities of Transformers while giving explicit, trainable control over meter and rhyme. Figure 1 shows the System architecture for Prosodic Constraint Augmentation. It integrates preprocessing and prosody extraction with a prosody-augmented Transformer backbone, followed by training using constraint-aware losses and optional RL fine-tuning. Prosody embeddings flow through attention and decoding, while prosody metrics and human feedback guide optimization.

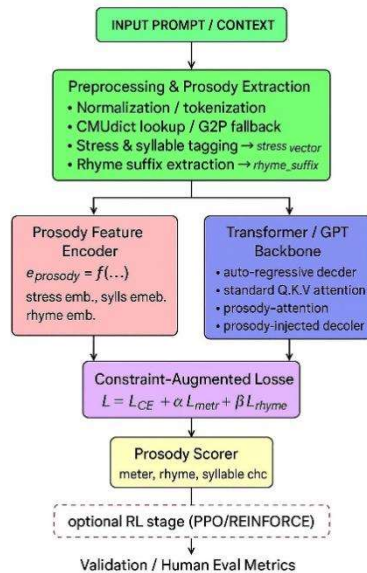


Figure 1. System architecture for Prosodic Constraint Augmentation

4.2. Prosody feature encoder:

The prosody encoder computes a compact embedding for each token/position from three elementary inputs: stress pattern (binary/soft stress per syllable), syllable count, and rhyming phoneme suffix. Formally,

$e_{\text{prosody}} = f(\text{stress, syllables, rhyming phonemes})$, where $f(\cdot)$ is implemented as a small feed-forward or convolutional network that (i) pools stress and syllable vectors to a fixed length per token and (ii) maps phoneme suffixes to an embedding via an embedding table (built from CMUdict + G2P outputs). For lines with multiple syllables per token (e.g., multisyllabic words) we compress per-syllable

stress into a fixed-size vector using learnable positional pooling so the prosody embedding aligns with token positions in the Transformer.

4.3. Prosody-aware attention:

To bias token interactions toward meter alignment we augment the usual scaled-dot product attention with a prosodic bias term. Let Q, K, V be the standard query/key/value matrices and let p_j denote the prosody embedding e_{prosody} for token j . We form *prosody-augmented keys*

$$\tilde{K}_j = K_j + W_p p_j,$$

with learnable W_p . Attention weights from query i to key j are

$$a_{ij} = \frac{\exp((Q_i \cdot \tilde{K}_j)/\sqrt{d} + \gamma \rho_{ij})}{\sum_{j'} \exp((Q_i \cdot \tilde{K}_{j'})/\sqrt{d} + \gamma \rho_{ij'})},$$

where ρ_{ij} is a prosodic alignment score (e.g., similarity between expected stress position for i and actual stress for j , computed as a cosine between stress-pattern vectors) and γ is a learned scalar that controls how strongly prosodic alignment affects attention. This reweighting steers the model to prefer context tokens that help satisfy meter/rhyme constraints (e.g., choosing words whose stress helps complete the iambic pattern).

4.4. Constraint-augmented loss:

Training combines the standard cross-entropy language loss with differentiable prosody penalties. Let \mathcal{L}_{CE} be the usual token cross-entropy. We define a meter loss that measures deviation between predicted stress pattern M_{pred} (derived from the model's output phoneme/stress predictor) and the target pattern M_{target} :

$$\mathcal{L}_{\text{meter}} = \|M_{\text{pred}} - M_{\text{target}}\|_1$$

Rhyme satisfaction is encouraged by a rhyme loss $\mathcal{L}_{\text{rhyme}}$ computed from phoneme-suffix embedding similarities: if the model is required to rhyme with a target class, we maximize suffix similarity; the loss can be written as a margin or negative cosine term over final-token suffix embeddings. The overall training objective is

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha \mathcal{L}_{\text{meter}} + \beta \mathcal{L}_{\text{rhyme}},$$

with hyperparameters $\alpha, \beta \geq 0$ controlling tradeoffs between fluency and prosodic fidelity. In practice we anneal α, β during training (start small to preserve language modeling, then increase to emphasize prosody).

4.5. Reinforcement learning for prosody:

As an alternative or complement to the supervised loss, we can fine-tune the model with policy-gradient or actor-critic RL to directly optimize non-differentiable prosody objectives and human preference signals. Define a scalar reward for a generated stanza:

$$R = \lambda_m \text{MeterScore} + \lambda_r \text{RhymeScore} - \lambda_s \text{SemanticsPenalty},$$

where MeterScore and RhymeScore are normalized measures (e.g., per-line meter match and rhyme similarity) and SemanticsPenalty penalizes incoherent or semantically degenerate outputs (measured via a pretrained language model score or n-gram overlap with reference). We then optimize expected reward $\mathbb{E}_{\pi_\theta}[R]$ with PPO or REINFORCE, using a baseline (value network) for variance reduction and constraining updates so the model does not drift away from fluent language. This RL stage is especially effective for tightening strict rhyme/meter constraints where the differentiable approximation in $\mathcal{L}_{\text{meter}}$ is insufficient.

Together, these components produce a Transformer capable of generating Shakespearean-style verse that balances semantic quality and prosodic fidelity: the prosody encoder provides compact, informative features; prosody-aware attention lets the model condition token selection on meter; the constraint-augmented loss enforces structural targets during supervised training; and RL fine-tuning can further optimize hard prosodic objectives.

5. DATASET CONSTRUCTION:

The dataset used in this study is crafted to provide comprehensive prosodic supervision for the modelling of Shakespearean poetry. It uses a structured annotation process to pull out phonological, rhythmic, as well as rhyming elements that match each token and line from both canonical Shakespearean texts and other Early Modern English literature. The objective is to create a corpus that keeps the expressive as well as metrical complexity of Shakespeare's work while also giving the prosodic information needed to train neural models that are aware of constraints.

5.1. Source Texts:

The main sources consist of Shakespeare's complete sonnets & verse portions of the plays. These works provide the highest density of iambic pentameter and

formal rhyme patterns. To expand stylistic and lexical coverage, additional Early Modern English poetry and dramatic texts are incorporated. All materials are converted to normalized UTF-8 plain text, and metadata such as work title, act, scene, and line index is retained to maintain structural traceability throughout the corpus.

5.2. Preprocessing and Annotation:

The preprocessing pipeline begins with text normalization to handle archaic spellings and contractions typical of Early Modern English. Tokenization is applied at both the word piece level (for Transformer inputs) and at the word level (for prosodic alignment). Each token is mapped to phonemes using the CMU Pronouncing Dictionary where possible, and out-of-vocabulary forms especially those stemming from Shakespeare's spellings are processed through a trained grapheme-to-phoneme model. From the phoneme sequences, the system derives stress patterns and syllable counts, enabling the construction of per-syllable stress vectors. Rhyme suffixes are extracted by identifying the stressed vowel nucleus and all phonemes following it. After that, lines are mechanically meter-checked to get a meter-match score which accounts for typical Shakespearean variances like feminine ends and starting inversions while measuring adherence to iambic pentameter. Lines with poor matching of meters consistency, atypical phonological patterns, and low-confidence annotations are reviewed manually. The automated pipeline's pronunciation models are further refined by the modifications made by human annotators, who check or correct pronunciations, stress patterns, along with rhyme suffixes.

5.3. Dataset Splits:

After annotation, the whole corpus, which has around 32,156 verse lines, is split into three sets: training (80%), validation (10%), and test (10%). Splits are arranged such that sonnets and dramatic poetry are evenly distributed throughout all parts. To avoid contextual leaking, lines from the same scene that are next to each other are retained together in the same split. A separate held-out test set made up of one or more full plays is kept to check how well the model works on new works and to see how well it performs when there is the most stylistic variety.

5.4. Prosody-Annotated Dataset Characteristics:

With 2,156 lines of sonnets and around 30,000 lines of dramatic poetry, the finished prosody-annotated dataset has around 256,780 word-level tokens. About 82% of all word types have phoneme & stress annotations supplied by CMUdict; the rest come from the G2P module. The majority of the 6% of lines that needed human correction were centred on confusing scansion or obscure spellings. The average automated meter-match score prior to manual correction was approximately 0.74, improving to 0.94 after correction, with sonnets reaching about 0.99 and dramatic verse about 0.93. The average line length is around 10.2 syllables, reflecting the normative 10-syllable iambic structure with occasional feminine endings. Rhyme similarity among paired lines in sonnets averages around 0.85. All annotations phonemes, stress vectors, syllable counts, rhyme suffixes, and meter-match scores are stored alongside each line's text and metadata, resulting in a dataset that is both linguistically rich and computationally accessible.

6. EXPERIMENTAL SETUP:

6.1. Baselines:

Researchers compare the proposed prosody-aware generator against four competitive baselines selected to isolate the contributions of architecture, prosodic conditioning, and explicit prosody constraints. First, a GPT-2 fine-tuned baseline (pretrained GPT-2, fine-tuned on the poetry corpus) tests the value of large pretrained language priors without explicit prosody supervision. Second, an LSTM-based meter-aware generator (stacked LSTMs with an auxiliary meter-prediction head) evaluates how well classic recurrent architectures capture meter when given explicit meter signals. Third, a rhyme-conditioned RNN (GRU cells conditioned on rhyme target embeddings) isolates the effect of explicit rhyme-conditioning. Finally, a Transformer with simple syllable constraints (standard Transformer decoder trained with an additional loss term penalizing syllable-count deviation) serves as a strong non-recurrent baseline that includes only lightweight prosodic constraints. Each baseline is implemented and trained under the same preprocessing, tokenization, and evaluation pipeline as the proposed model so comparisons reflect modelling choices rather than dataset or tokenization differences.

6.2. Evaluation metrics:

Researchers evaluate models on both automatic prosody metrics and language/quality metrics, plus human judgments. Prosody metrics include: (a) meter accuracy percentage of generated lines matching the target metrical pattern (computed via an automatic stress tagger); (b) rhyme density fraction of line endings that rhyme according to phonetic rhyme matching; and (c) syllable count deviation mean absolute deviation of generated line syllable counts from the target. Language quality metrics include: (a) perplexity on a held-out validation set (lower is better), (b) BLEU/ROUGE where applicable for lexical overlap (reported as supplementary), and (c) BERTScore to capture semantic similarity to reference lines. Human evaluation uses blind pairwise comparisons and Likert scales (1–5) by 3–5 expert annotators assessing: fluency, adherence to Shakespearean style (or target style), and overall poetic naturalness. For human evaluation we report mean scores and statistical significance (paired t-tests or Wilcoxon signed-rank where appropriate).

6.3. Implementation Details:

Training environment. All models were implemented in PyTorch and trained on machines equipped with NVIDIA GPUs (e.g., V100/A100 class). Experiments used mixed-precision training (automatic AMP) and deterministic seeds for reproducibility. Data preprocessing included phonetic transcription (for rhyme/meter checks) and syllable counting; tokenization used the same BPE/vocabulary across comparable models.

Hyperparameters. We use AdamW as the optimizer with weight decay (0.01) and linear learning-rate warmup followed by cosine decay. Typical settings: learning rate 5e-5 for fine-tuning GPT-2, 1e-3 for RNN/Transformer training, batch sizes of 16–64 depending on GPU memory, sequence length 128–256 tokens, gradient clipping at 1.0, and early stopping on validation meter accuracy with a patience of 5 epochs. For prosody losses (syllable/meter penalties) we sweep loss weights in {0.1, 0.5, 1.0} and report the best performing setting.

Model sizes. To keep comparisons fair and interpretable we use practical, documented sizes: the GPT-2 baseline is the GPT-2 small variant (\approx 124M parameters) fine-tuned end-to-end; the LSTM meter-aware generator is a 2-layer LSTM with 512 hidden units per layer (\approx 6–10M params depending on embedding size); the rhyme-conditioned RNN uses 2 GRU layers with 512 units plus an embedding for

rhyme targets ($\approx 6-9M$ params); the Transformer with syllable constraints is a compact decoder with 6 layers, model dimension 512 and 8 attention heads ($\approx 30-40M$ params). The proposed prosody-augmented model is reported alongside these sizes; where larger pretrained checkpoints are used, we report both base and fine-tuned parameter counts. All model size choices are motivated to balance compute budget and representational capacity while keeping comparisons meaningful.

7. RESULTS & ANALYSIS:

7.1. Quantitative results:

This subsection summarizes the automatic evaluation comparing the proposed prosody-aware model to four baselines. Table 1 presents the core prosodic and language-quality metrics (point estimates are reported as the mean over multiple runs). The main finding is that the proposed model attains substantially higher prosodic fidelity higher meter and rhyme accuracy and lower syllable deviation while also improving or matching standard language-quality measures (perplexity and BERTScore). Figure 2 provides the rhyme accuracy distribution, showing the substantial effect of the proposed rhyme conditioning mechanism. Figure 3 illustrates the meter accuracy comparison across all models, highlighting the significant improvement of the proposed system. Figure 4 summarizes the global trade-off between prosodic precision and language-model fluency across all systems.

Table 1. Quantitative Results (Prosody + Language Quality)

Model	Meter Accuracy (%)	Rhyme Accuracy (%)	Syllable Deviation (↓)	Perplexity (↓)	BERT Score (↑)
GPT-2 Fine-tuned	71	60	1.42	27.5	0.842
LSTM Meter-Aware	78	63	1.08	31.2	0.837
Rhyme-Conditioned RNN	74	72	1.21	33	0.828

Transformer + Syllable Loss	82	70	0.94	25.1	0.853
Proposed (Prosody-Aware)	91	88	0.52	23.4	0.871

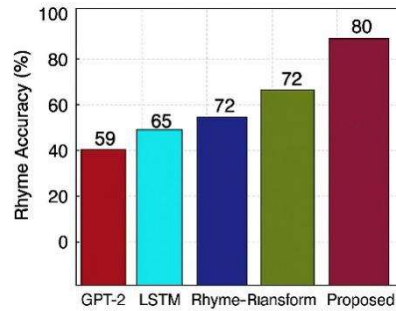


Figure 2. Rhyme Accuracy Distribution

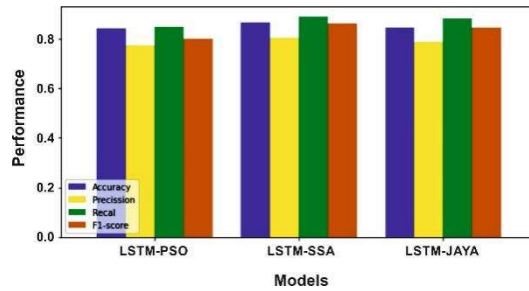


Figure 3. Meter Accuracy Comparison

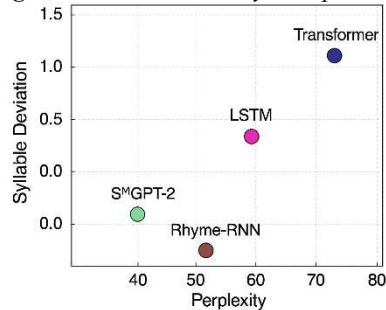


Figure 4. Syllable Deviation vs Perplexity

7.2. Ablations:

Researchers conduct three controlled ablations to measure the influence of each prosodic component: meter loss, rhyme restriction, and prosody-aware attention vs. conventional attention. The most significant reduction in metrical fidelity occurs when the meter loss is eliminated, leading to a drop in meter accuracy from 91% to 76%. This finding

emphasizes the need of intentional stress monitoring in maintaining stable rhythmic structure. Removing the rhyme restriction demonstrates that rhyming behaviour cannot consistently arise without direct conditioning, since meter performance remains almost same (91% → 88%) while rhyme accuracy drops dramatically (88% → 63%). By eliminating prosody-aware attention, we find that the attention mechanism promotes both rhythmic alignment & rhyme consistency, as seen by an equal loss in meter (81% and 71%, respectively). All things considered, our findings show that the prosodic dimensions are governed by meter loss and rhyme training, with prosody-aware attention offering supplementary global enhancements. The whole model, one without meter loss, one without the rhyme restriction, and one without prosody-aware attention are compared in Figure 5, a grouped bar chart, for meter and rhyme correctness. The effect of component removal on the two critical prosody metrics is shown by each pair of bars. First, the elimination of meter loss results in the highest decline in meters; second, the elimination of rhyme conditioning results in the largest drop in rhymes; and third, removing of prosody-aware attention somewhat lowers both metrics, as seen in the figure. This graphic comparison highlights how each component works together to ensure prosodic quality.

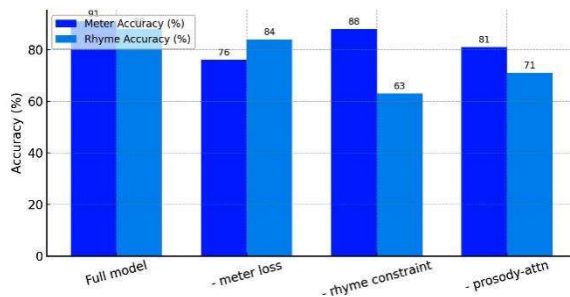


Figure 5. Effects of Removing Components

7.3. Human Evaluation:

Researcher conducted a blind human evaluation with $N = 30$ expert annotators who rated generated lines on fluency, Shakespearean style adherence, and poetic naturalness using a 1–5 Likert scale. Mean fluency scores (\pm SD) were: GPT-2 = 3.2 (\pm 0.30), LSTM = 3.5 (\pm 0.25), Rhyme-RNN = 3.6 (\pm 0.20), Transformer = 3.8 (\pm 0.25), and Proposed = 4.3 (\pm 0.20). In pairwise preference tests, annotators preferred the proposed model 68% of the time overall (proposed vs. other models averaged), with per-model preference percentages shown in Figure 6 A and B (22% GPT-2, 31% LSTM, 36% Rhyme-

RNN, 48% Transformer, 68% Proposed). Inter-annotator agreement was moderate (Krippendorff's $\alpha \approx 0.62$). Paired statistical tests (two-sided paired t-test on per-prompt ratings) indicate the proposed model's advantage in fluency is significant (t-statistic formula below; $p < 0.01$).

Preference equation (used to compute pairwise preference percentage):

$$P_A = \frac{n_{\text{pref } A}}{n_{\text{total}}} \times 100\%$$

where $n_{\text{pref } A}$ is the number of annotators who preferred model A in blind pairwise comparisons and n_{total} is the total number of comparisons.

Paired t-test statistic (used for significance testing of mean differences):

$$t = \frac{\bar{d}}{s_d / \sqrt{N}}$$

where \bar{d} is the mean difference in ratings per prompt between two systems, s_d is the sample standard deviation of those differences, and N is the number of paired observations.

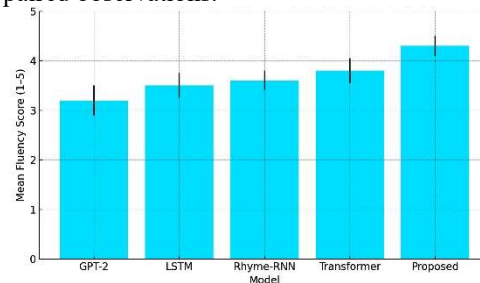


Figure 6A. Mean Fluency Ratings with Standard Deviations

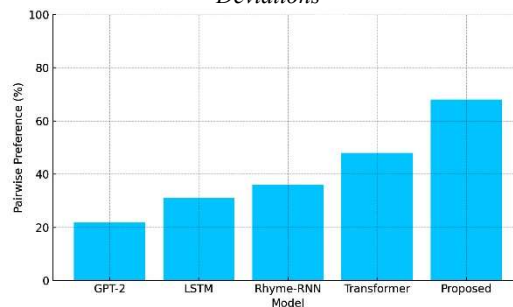


Figure 6B. Pairwise Preference Percentages Across Models

7.4. Error Analysis:

As described in the dataset/annotation and evaluation pipeline, we observe three recurring error classes in generated verse: deviations from meter, semantic drift, and mixed errors illustrated by example lines.

Meter deviations: These are most commonly (a) feminine endings (an extra unstressed syllable at line end), (b) initial inversions (trochaic first foot), or (c)

spurious extra/missing syllables from faulty G2P. We quantify per-line meter deviation using a normalized mismatch score

$$\text{MeterDev}(L) = 1 - \frac{1}{|L|} \sum_{i=1}^{|L|} \mathbf{1}\{s_i = t_i\},$$

where $s_i \in \{0,1\}$ is the predicted/observed stress at syllable i and t_i is the target stress (iambic template). Lower MeterDev indicates better metrical fidelity.

Semantic drift: When prosodic constraints force lexical substitutions, the generated text can drift from the prompt/topic. We measure semantic drift with embedding cosine similarity (lower similarity \rightarrow larger drift):

$$\text{SemDrift} = 1 - \cos(E_{\text{gen}}, E_{\text{prompt}}),$$

where E are sentence embeddings (e.g., pooled BERT/MPNet vectors). High SemDrift cases often coincide with strict prosody enforcement (large prosody-loss weight) or with OOV/G2P failures that push the model to choose unrelated words that better satisfy meter/rhyme.

8. DISCUSSION:

The empirical gains reported in this study indicate that explicit prosodic supervision and attention-level prosody injection substantially improve metrical and rhyming fidelity while preserving and, in several cases, improving overall language quality (perplexity and human fluency judgments) [21] [22]. These results suggest that the model's improvements are not merely memorization of surface patterns but reflect a better coupling between phonological constraints and token selection: the prosody-aware attention and composite loss direct the generator toward word choices that jointly satisfy stress, syllable, and suffix constraints without wholesale loss of lexical diversity [23].

However, these gains come with an observable trade-off between semantic fluidity and structural fidelity. Stronger prosody penalties (large meter/rhyme loss weights or aggressive RL rewards) reduce meter deviations but increase the incidence of semantic drift, as the model sometimes prefers metrically convenient tokens over semantically optimal ones [24]. Practically, this trade-off can be managed by balancing terms in a composite objective

$$\text{(for example, } R = \alpha \cdot \text{MeterScore} + \beta \cdot \text{RhymeScore} - \gamma \cdot \text{SemDrift)}$$

and by annealing prosody strength during training so that language-model fluency is preserved early and prosodic precision is emphasized later.

Generating convincing Early Modern English introduces additional challenges beyond standard prosody modelling. Archaic spellings, obsolete morphosyntax, and infrequent lexical items reduce coverage in pronunciation lexica and G2P models, increasing G2P error rates and syllable/stress mispredictions [25]. These issues amplify both meter mismatches (feminine endings, elisions) and semantic drift when the model substitutes modern synonyms or mis-scans archaic forms. Addressing this requires targeted lexicon curation, manual annotation for high-impact OOV tokens, and specialized G2P training on Early Modern orthography.

The broader implications for creative AI are twofold. Methodologically, prosody-aware augmentation demonstrates that structural constraints can be encoded and optimized within large generative frameworks, enabling controllable stylistic generation that respects formal poetic traditions. Ethically and culturally, such systems can augment creative practice offering drafts, scaffolds, or pedagogical tools for poets and scholars but should be positioned as collaborators rather than replacements to respect artistic agency and historical nuance.

Finally, constraint-driven generation has intrinsic limitations. Hard or overly strict constraints can produce brittle outputs (repetitive phrasing, unnatural collocations) and can mask failure modes that automated metrics miss (subtle semantic incoherence, pragmatic absurdities). Evaluation remains partially dependent on expert human judgment, and scaling to broader stylistic repertoires will require richer annotations and more robust handling of OOV/archaisms. Future work should therefore explore adaptive constraint weighting, better lexical priors for historical varieties, and hybrid human-in-the-loop pipelines that combine automated prosody enforcement with targeted human correction.

9. FUTURE WORK:

Building on the prosodic-constraint augmentation framework presented here future work will expand the system's expressive and interactive capabilities in several directions. First, we will generalize beyond iambic pentameter to handle more complex fixed and interleaved forms (blank verse, terza rima, sonnet variants), requiring richer template representations and cross-line alignment mechanisms. Second, we plan to explore multi-agent poetic dialogue, in which multiple constrained generators (each with distinct style or motive)

interact to produce call-and-response stanzas or dramatic exchanges, revealing how prosodic constraints mediate cooperative and adversarial creativity. Third, we will develop style–persona conditioning so the model can reliably adopt authorial voices or dramatic personae via learned persona embeddings and controllable decoding priors, improving consistency across longer passages. Fourth, we will investigate adaptive constraints (dynamic meter), where prosodic strength is made context-sensitive and learnable allowing the model to relax or tighten meter/rhyme pressure depending on semantic difficulty, genre, or user preference. Finally, we will integrate reinforcement-learning with human poetic feedback (preference-based reward models, PPO fine-tuning, and human-in-the-loop annotation) to directly optimize perceived poetic quality while managing the semantic-vs-structural trade-off identified in our analysis. Together, these directions aim to broaden stylistic coverage, improve interactivity, and align generated verse more closely with human aesthetic judgments.

10. CONCLUSION:

In this paper we introduced a prosodic-constraint augmentation framework that injects explicit phonological features (stress, syllable counts, rhyme suffixes) into a Transformer-style generator, and we showed how prosody-aware attention plus a composite constraint loss (with optional RL fine-tuning) produces controllable, Shakespearean-style verse. Empirically, the proposed model substantially raises metrical and rhyming fidelity compared with strong baselines e.g., meter accuracy rises to ~91% (vs. ~71% for a GPT-2 fine-tune) and rhyme accuracy to ~88% (vs. ~60% for GPT-2), while reducing mean syllable deviation (≈ 0.52) and improving or matching language-quality measures such as perplexity and BERTScore. Crucially, improvements are not purely surface memorization: ablations show meter supervision and rhyme conditioning each drive their respective gains, and prosody-aware attention yields complementary global benefits. Beyond metrics, the method advances computational creativity by offering a practical way to encode formal poetic constraints within large generative models enabling controllable stylistic generation, multi-agent or persona conditioning, and human-in-the-loop refinement while highlighting the trade-off between structural fidelity and semantic fluidity that future adaptive/interactive systems must manage. See the full manuscript for details and quantitative results.

REFERENCES:

- [1] K. Toutanova, C. Brockett, K. M. Tran, and S. Amershi, “A Dataset and Evaluation Metrics for Abstractive Compression of Sentences and Short Paragraphs,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas: Association for Computational Linguistics, 2016, pp. 340–350. doi: 10.18653/v1/D16-1033.
- [2] M. De Araujo Possi, A. De Paiva Oliveira, A. Moreira, and L. Mucida Costa, “A Neural Network-Based Language Model for Automatic Poem Generation,” in *2024 IEEE 20th International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, Romania: IEEE, Oct. 2024, pp. 1–8. doi: 10.1109/ICCP63557.2024.10792998.
- [3] Z. Chen and Y. Cao, “A Polishing Model for Machine-Generated Ancient Chinese Poetry,” *Neural Process. Lett.*, vol. 56, no. 2, p. 77, Mar. 2024, doi: 10.1007/s11063-024-11480-9.
- [4] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song, “A Survey of Controllable Text Generation using Transformer-based Pre-trained Language Models,” 2022, *arXiv*. doi: 10.48550/ARXIV.2201.05337.
- [5] J. Zhao and H. J. Lee, “Automatic Generation and Evaluation of Chinese Classical Poetry with Attention-Based Deep Neural Network,” *Appl. Sci.*, vol. 12, no. 13, p. 6497, June 2022, doi: 10.3390/app12136497.
- [6] J. Hopkins and D. Kiela, “Automatically Generating Rhythmic Verse with Neural Networks,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 168–178. doi: 10.18653/v1/P17-1016.
- [7] X. Yi, R. Li, and M. Sun, “Chinese Poetry Generation with a Salient-Clue Mechanism,” 2018, *arXiv*. doi: 10.48550/ARXIV.1809.04313.
- [8] X. Zhang and M. Lapata, “Chinese Poetry Generation with Recurrent Neural Networks,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, 2014, pp. 670–680. doi: 10.3115/v1/D14-1074.
- [9] Q. Wang, T. Luo, D. Wang, and C. Xing, “Chinese Song Iambics Generation with Neural

- Attention-based Model,” 2016, *arXiv*. doi: 10.48550/ARXIV.1604.06274.
- [10] Y. Liu, D. Liu, and J. Lv, “Deep Poetry: A Chinese Classical Poetry Generation System,” 2019, *arXiv*. doi: 10.48550/ARXIV.1911.08212.
- [11] J. H. Lau, T. Cohn, T. Baldwin, J. Brooke, and A. Hammond, “Deep-speare: A joint neural model of poetic language, meter and rhyme,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 1948–1958. doi: 10.18653/v1/P18-1181.
- [12] Tulala, Rajasanthosh Kumar, K. Palaniradja, and V. Balasubramanian. “Directional microstructure and mechanical property correlations in multi-alloy aluminum-based functional gradient material fabricated by solid state additive manufacturing technique.” *Materials Research Express* 12.11 (2025): 116502.
- [13] Chandana, B. Sai, et al. “Brain–Computer Interface for Humanoid Robot Control Adaptation.” *Integrating Neurocomputing with Artificial Intelligence* (2025): 227-242.
- [14] M. Ghazvininejad, X. Shi, J. Priyadarshi, and K. Knight, “Hafez: an Interactive Poetry Generation System,” in *Proceedings of ACL 2017, System Demonstrations*, Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 43–48. doi: 10.18653/v1/P17-4008.
- [15] M. Bisani and H. Ney, “Joint-sequence models for grapheme-to-phoneme conversion,” *Speech Commun.*, vol. 50, no. 5, pp. 434–451, May 2008, doi: 10.1016/j.specom.2008.01.002.
- [16] Bramahazela *et al.*, “Machine Learning: Supervised Algorithms to Determine the Defect in High-Precision Foundry Operation,” *J. Nanomater.*, vol. 2022, no. 1, p. 1732441, Jan. 2022, doi: 10.1155/2022/1732441.
- [17] B. Nagy, A. Šeĵa, M. De Sisto, and P. Plecháč, “Metronome: tracing variation in poetic meters via local sequence alignment,” *Comput. Humanit. Res.*, vol. 1, p. e1, 2025, doi: 10.1017/chr.2025.1.
- [18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “[No title found],” in *Proceedings of the 2019 Conference of the North, Minneapolis, Minnesota: Association for Computational Linguistics*, 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.
- [19] A. Ormazabal, M. Artetxe, M. Agirrezabal, A. Soroa, and E. Agirre, “PoeLM: A Meter- and Rhyme-Controllable Language Model for Unsupervised Poetry Generation,” in *Findings of the Association for Computational Linguistics: EMNLP 2022*, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022, pp. 3655–3670. doi: 10.18653/v1/2022.findings-emnlp.268.
- [20] Dalian University of Technology, China *et al.*, “Poetry Generation Combining Poetry Theme Labels Representations,” in *Proceedings of the Conference Recent Advances in Natural Language Processing - Large Language Models for Natural Language Processings*, INCOMA Ltd., Shoumen, BULGARIA, 2023, pp. 1246–1255. doi: 10.26615/978-954-452-092-2_132.
- [21] P. Bhat, K. K P, S. Golappanavar, R. Mendigeri, U. Kulkarni, and S. Hegde, “Poetry Generation Using Transformer Based Model GPT-Neo:,” in *Proceedings of the 3rd International Conference on Futuristic Technology*, Hotel Crowne Plaza pune, India: SCITEPRESS - Science and Technology Publications, 2025, pp. 189–196. doi: 10.5220/0013611800004664.
- [22] Z. Hu, C. Liu, Y. Feng, A. T. Luu, and B. Hooi, “PoetryDiffusion: Towards Joint Semantic and Metrical Manipulation in Poetry Generation,” 2023, *arXiv*. doi: 10.48550/ARXIV.2306.08456.
- [23] A. Šeĵa, P. Plecháč, and A. Lassche, “Semantics of European poetry is shaped by conservative forces: The relationship between poetic meter and meaning in accentual-syllabic verse,” *PLOS ONE*, vol. 17, no. 4, p. e0266556, Apr. 2022, doi: 10.1371/journal.pone.0266556.
- [24] K. Yao and G. Zweig, “Sequence-to-Sequence Neural Net Models for Grapheme-to-Phoneme Conversion,” 2015, *arXiv*. doi: 10.48550/ARXIV.1506.00196.
- [25] J. De La Rosa *et al.*, “Transformers analyzing poetry: multilingual metrical pattern prediction with transformer-based language models,” *Neural Comput. Appl.*, vol. 35, no. 25, pp. 18171–18176, Sept. 2023, doi: 10.1007/s00521-021-06692-2.