Please cite the Published Version

Tabassam, Muhammad Rauf , Waheed, Hajra , Safder, Iqra , Sarwar, Raheem , Aljohani, Naif Radi , Nawaz, Raheel , Hassan, Saeed-Ul , Zaman, Farooq and Ahsan, Ahtazaz (2024) UPON: Urdu Poetry Generation Using Deep Learning: A Novel Approach and Evaluation. ACM Transactions on Asian and Low-Resource Language Information Processing. ISSN 2375-4702

UPON: Urdu Poetry Generation Using Deep Learning: A Novel Approach and Evaluation

MUHAMMAD RAUF TABASSAM, Information Technology University, Lahore, Pakistan

HAJRA WAHEED, School of Computing, National University of Computer and Emerging Sciences, Lahore, Pakistan

IQRA SAFDER, School of Computing, National University of Computer and Emerging Sciences, Lahore, Pakistan

RAHEEM SARWAR, OTEHM, School of Business and Law, Manchester Metropolitan University - All Saints Campus, Manchester, United Kingdom of Great Britain and Northern Ireland

NAIF RADI ALJOHANI, King Abdulaziz University, Jeddah, Saudi Arabia

RAHEEL NAWAZ, Staffordshire University, Stoke-on-Trent, United Kingdom

SAEED-UL HASSAN, Manchester Metropolitan University, Manchester, United Kingdom

FAROOQ ZAMAN, Information Technology University, Lahore, Pakistan

AHTAZAZ AHSAN, Information Technology University, Lahore, Pakistan

Poetry represents the oldest and most esteemed literary form, allowing poets to convey ideas while carefully attending to elements such as meaning, coherence, poetic quality, and fluency. Notably, the creation of good poetry entails considerations of rhyme and meter. With the advent of artificial intelligence (AI), significant advancements have been made in automatic text generation, primarily within languages such as English and Chinese. However, the generation of Urdu poetry presents a unique challenge due to the language's inherent ambiguity, cultural and historical nuances, and the demand for creativity. The existing body of literature has only marginally explored Urdu prose and has almost entirely overlooked the domain of Urdu poetry generation, primarily due to the scarcity of comprehensive training data. In response to this deficiency, this research endeavor addresses this challenge. It begins by introducing a specialized Urdu poetry dataset adhering to a specific meter, 'behr-e-khafeef,' which incorporates approximately 20,000 couplets from the Rekhta repository. Subsequently, a character-based encoding methodology is proposed to transform these couplets into a numerical representation, assigning a distinct identifier to each character. The generation process initiates with the creation of the first verse through a character-level LSTM, followed by the application of a machine translation technique, specifically sequence-to-sequence learning, to formulate the second verse based on the first. The generated poetry is subjected to evaluation based on metrics, including BLEU scores. Additionally, an expert panel of Urdu poets is engaged to conduct a human assessment of the generated couplets, with the evaluation encompassing critical dimensions such as meaning, coherence, poetic quality, and fluency. Our

Authors' Contact Information: Muhammad Rauf Tabassam, Information Technology University, Lahore, Punjab, Pakistan; e-mail: rauf.tabassam@itu.edu.pk; Hajra Waheed, School of Computing, National University of Computer and Emerging Sciences, Lahore, Punjab, Pakistan; e-mail: hajra.waheed@nu.edu.pk; Iqra Safder, School of Computing, National University of Computer and Emerging Sciences, Lahore, Punjab, Pakistan; e-mail: iqrasafder@gmail.com; Raheem Sarwar, OTEHM, School of Business and Law, Manchester Metropolitan University - All Saints Campus, Manchester, United Kingdom of Great Britain and Northern Ireland; e-mail: R.Sarwar@mmu.ac.uk; Naif Radi Aljohani, King Abdulaziz University, Jeddah, Saudi Arabia; e-mail: nraljohani@kau.edu.sa; Raheel Nawaz, Staffordshire University, Stoke-on-Trent, Staffordshire, United Kingdom; e-mail: raheel.nawaz@staffs.ac.uk; Saeed-Ul Hassan, Manchester Metropolitan University, Manchester, Manchester, United Kingdom; e-mail: saeedulhassan@gmail.com; Farooq Zaman, Information Technology University, Lahore, Pakistan; e-mail: phdcs18002@itu.edu.pk; Ahtazaz Ahsan, Information Technology University, Lahore, Pakistan; e-mail: ahtazaz.ahsan@itu.edu.pk.

findings are juxtaposed with existing poetry generation systems, demonstrating a notable advancement in the state-of-the-art, as evidenced by a BLEU score of 0.23. The research culminates with the presentation of prospective avenues for further exploration, aimed at inspiring the scholarly community to enhance the domain of poetry generation and augment existing contributions in this field.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; *Natural language processing*; Language resources.

Additional Key Words and Phrases: UPON, LSTM, GRU, BLEU, Poetry generation, Seq2Seq, Encoder, Decoder, Human Evaluation

1 Introduction

Literature serves as the embodiment of a language's identity, encompassing diverse forms such as poetry, prose, drama, and folktales. Among these, poetry stands out as the most enchanting and esteemed type of literary expression. It relies on the interplay of words and rhythmic patterns, serving as a captivating medium to convey messages within concise lines. Given its multitude of genres, structures, and forms, each with its distinct characteristics, defining poetry concretely and comprehensively proves challenging. The concept of poetry remains highly personalized, much like other forms of creative expression in literature, defying a universally exhaustive definition. One can easily encounter a piece of poetry that defies any established definition, thus highlighting its inherent fluidity and individuality. The Oxford English dictionary defines poetry as "composition in verse or metrical language, or in some equivalent patterned arrangement of language; usually also with the choice of elevated words and figurative uses, and the option of a syntactical order, differing more or less from those of ordinary speech or prose writing" [49]. A Greek philosopher, Aristotle described poetry as "poetry is more philosophical and of higher value than history; for poetry tends to express the universal, history the particular" [72]. Another definition of poetry by Paul Engle says that "poetry is boned with ideas, nerved and blooded with emotions, all held together by the delicate, tough skin of words" [72]. It is tough to define poetry, yet most people identify it when they see it. Poetry, specifically in the form of songs, predates prose as a means of emotional expression. It has a deep connection to human emotions, encompassing a wide range of feelings including happiness, sorrow, skepticism, hope, loneliness, and love, among others. An excellent example of this is found in patriotic songs, which serve as poetic expressions that evoke strong emotions such as joy, triumph, and pride for one's country and its heroes [72].

The following are some key terms of poetry.

- Verse (مصرع): A single line in a poetic composition is called verse. Verse represents a group of words in a poetic composition.
- Stanza (قطعہ): The grouping of verses is called a stanza.
- Couplet (شعر): A group of two verses of the same meter is called a couplet. It is a unit for a poem or ghazal.
- Rhyme (قافیہ): A rhyme is a word that has similar sounds, most frequently in the last word of a verse in poetry or a song. It is located before the refrain in poetry.
- Refrain (ردیف): Refrains are the words repeated after the rhyme in a poem.
- Meter (بحر): Meter refers to the pattern of sounds that vary the stressed/long, and unstressed/short syllables. This syllable pattern remains the same in the poem.
- Syllable: It is a unit of speech sounds. It can be defined as the sound of a vowel or vowel with a consonant when pronouncing a word. E.g., the word hotel is formed with two syllables: "ho" and "tel".
- Meter Foot: Meter foot is the syllable sequence used in a meter. There can be multiple meter feet for a single meter.

Meter (called Behar (بحر) in Urdu) is a specific syllable pattern a verse must follow. A long syllable is denoted by an equal symbol (=), number digit 2, or with the letter L. Similarly, the short syllable is denoted by a dash symbol (-), digit 1, or the letter S. In this research, the letters L and S are used for long and short syllables

respectively. The syllable pattern of the meter considered in this research is LSLL SLSL LL. In Urdu, this meter name is Behar-e-Khafeef (خفیف بحرِ), and meter foot is فعلن مفاعلن فاعلاتن. In this meter, the last two syllables can have multiple variations. It can be LL (فعلن) or SSL(فَعِلن). Adding an extra short syllable at the end of the meter called Izafat is allowed in Urdu poetry. Hence last two syllables can be LL (فعلن), SSL (فَعِلن), LLS (مفعول), or SSLS (فَعِلات).

There is no perfect definition of poetry, so to measure the automated generated poetry lines as a poem, it is imperative to define some properties. These properties will then determine the proximity of a sentence generated by the system to a valid verse. Poetic properties of poetry are Grammaticality, Meaningfulness, and Poeticness [32].

- Grammaticality: A verse or poem must follow the language conventions defined by the grammar and lexicon. However, poetry is less constrained compared to conventional text. This property removes the random sequences from the generated text.
- Meaningfulness: A text should intentionally convey a meaningful and conceptual message. This is a general property that does not hold for just poetry but for all types of texts. However, this property must be satisfied for some text to be poetry.
- Poeticness: It refers to the poetic features such as meter and rhyme. A poem must exhibit these features to be called poetry. It should follow the defined rhythmic patterns.

Considering these properties, we can define poetry as a natural language artefact that fulfils the above-defined properties at the same time. Alternatively, a text t is poetry if $t \in Grammaticality \cap Meaningfulness \cap Poeticness$.

The distinctive contributions of this research are as follows.

(1) A poetry dataset is compiled focusing on a specific meter, consisting of over 20, 000 couplets from over 1, 000 poets.
(2) A meter-specific poetry generation system is proposed, using two different models for first and second verse generation. We used the LSTM model for the first verse and a sequence-to-sequence language translation model for the second verse generation.
(3) For evaluating our proposed approach, BELU score and human evaluation are employed to assess the deep learning models.

2 Literature Review

Applications of AI can be found across various domains, and it has the potential to impact numerous industries. Some notable domains where AI applications are prevalent include: Healthcare, Fraud Detection, Education, Retail, Autonomous Vehicles, Natural Language Processing (NLP), Cybersecurity, Manufacturing, Entertainment, and Robotics [20, 22, 28, 29, 56, 58–64, 71].

The utilization of computer programs to autonomously generate artistic creations has witnessed a significant rise in recent years, leading to the recognition of computers as artists in their own right. Actual creative systems work in a variety of disciplines, including the production of creative literature, music composition, and visual art. The generation of human language is a well-known and potential area of computational linguistics and artificial intelligence. Its fundamental objective is to create programming code that can generate understandable text. Examples of the many kinds of automatically created content include weather forecasting [7], autobiographies [24], and text with creative elements. Algorithms that can generate poetry, humor, or story narrative [8, 15] and [18] are examples of automatic creative text generation efforts. This section discusses the recent poetry generation models based on different languages, poetry features, AI methods, and evaluation approaches.

People have written poetry in different languages, and because every language is unique, the poetic culture is centred on various forms of each language. Mainly poetry generation systems focus on English and Chinese, but some work is also available in other languages. The earliest poetry generation experiments were done in French [6, 50, 53]. However, Spanish was among the first languages where this problem was studied in the

framework of AI [16, 17, 48]. Song lyrics in Portuguese were created automatically for particular music [43], and poetry was created using user-provided structures, including the stanzas, syllables per stanza, rhyme scheme, and lines. [39]. The same architecture for poetry generation is used for different languages like Spanish and English. [42]. Spanish poetry was also produced using another poem generator that was initially created for English [19]. Another technique for poetry generation used transformers for Spanish poetry generation [44]. Poetry generation in the Afrikaans language, AfriKI, is also attempted with a small corpus in [74].

Asian languages were also explored, including some with different rhythm and rhyme requirements in poetry. For example, the creation of Tamil music lyrics [3, 13, 21, 25, 54], Bengali poetry lines generation that fit the beat of a user-provided line [12], Indonesian poetry by taking inspiration from the news [51], and traditional Chinese poetry with a specific tone, rhythm criteria, and metrical constraints [30, 76, 77, 79]. Some latest work in poetry generation for the Arabic language was done by Sameerah and Banafsheh et al. [65]. This was marked as the pioneering study in the Arabic script style. Subsequently, further investigations were conducted into the generation of poetry in Arabic [5, 23]. Binari is the first ever poetry generation model in another Asian language, Turkish [78].

Systems for generating poetry may deal with a wide range of features, both formally and in terms of content. The form plays a crucial role in promptly recognizing the generated writing as poetry. A regular meter and rhyme are undoubtedly the most prominent form-related characteristics. These are pretty simple for computer programs to manage, specifically when contrasted with content features. A meter is often represented by the sequence of syllables in every verse, including rhymes and stress patterns. Gervas et al. [17] and Manurung et al. [32] characterize the stressed syllable positions and rhymes. Haikus, a type of Japanese poetry, consists of 3 lines having 5, 7, and 5 syllables respectively in each line [32, 37], and some current haikus with other frequency of syllables [75]. Limerick is a type of humorous poem composed of five verses that often have longer first, second, and fifth lines that rhyme with AABBA [26, 32]. The sonnet is a traditional type of poetry that consists of 14 verses, usually with ten syllables in each line. It may have various groupings, stress patterns, and rhyme systems [19]. Spanish traditional poetry forms "romance" have verses of eight syllables, and all even-numbered syllables are rhymed together. The "tercetos encadenados" have three verses of 11 syllables, and "cuarteto" has four verses of 11 syllables with two outer verses rhyming together. [16, 17]. Classical Basque poetry, known as "bertsolaritza", has specific meter and rhyme requirements [1]. Lewis et al. [27] trained a syllable neural model for poetry generation with the verse style of William Wordsworth.

Various rule-based approaches have been employed to handle meter and rhyme in Portuguese [43] and Spanish [17]. There are some systems designed to produce lyrics for songs that have less conventional structures but focus on the meter with other musical features. These systems include tunes that have beats with tense and weaker rhythms [25, 40, 43], or rap where in addition to rhyme, assonance is represented as the repeating vowel phonemes [31, 47].

In addition to being poetry, poetic writing should also include two other essential qualities, as per Manurung et al. [32]: it must be grammatically correct (grammaticality) and transmit a message that can be understood by the reader (meaningfulness). Many of the studied systems tend to follow syntactic norms to some extent because they depend on lexical-syntactic patterns, text segments, or linguistic models learned using the text written by humans. Meaningfulness, on the other hand, is more complex to manage automatically. Different systems process the input document in different methods for extracting meaningful information that may be utilized in the poetry. For example, Toivanen et al. [68] identify new links from well-known associations in the text. Tobing and Manurung et al. [67] use the dependency relations of text documents and apply them to define the generated poetry. In the last example, Gonçalo Oliveira and Alves et al. [41] create a semantic network using concept maps that are taken from the input text.

Terms that appear in similar scenarios have similar meanings, which focus on the way of using language in some document sets. To generate poetry, McGregor et al. [34] and Chun and Wong et al. [75] use vector space

models based on words. Yan et al. [76] learned the word embeddings from a dataset of poems, and Malmi et al. [31] used sentence word embeddings to utilize semantic features.

In some systems, the generation of text incorporates grammar, semantics, and sentiment aspects to control the syntax of the resulting text. In [39], the grammar and semantics are closely connected since every rule conveys some particular semantic connection that may apply to any word pair with that relation. Colton et al. [11] proposed a system that generates natural language comments for each created poem, giving some context for poetry generation. A similar property is discussed in [41] and [42]. Moreover, Misztal and Indurkhya et al. [36] populate the sentiment feature by using the WordNet effect.

The early poetry generators described in [6, 50] are entirely based on the combinatory operations applied to a collection of human poetry. Meanwhile, intelligent poetry generator systems apply semantics when choosing contents and apply computational approaches that enhance the process. It improves the generation process by allowing for an effective analysis of the domain of potential candidates. It makes it feasible to control additional characteristics, frequently with an eye towards a specified aim, and occasionally produce poems that are more innovative.

The case-based reasoning technique uses previous solutions in a four-step process of retrieve, reuse, revise, and retain, to solve new related problems. In poetry generation systems, Gravas et al. [17] and Diaz-Agudo et al. [14] used case-based reasoning techniques by retrieving vocabulary and verses, reusing them with the part of speech examples to generate new lines, revisioning the produced lines by performing linguistic analysis and retaining it to generate next elements.

Chart Parsing is a method of parsing text under context-free grammar by using dynamic programming. Chart Generation is a technique defined as the inversion of chart parsing and is utilized by some poem generation systems [32, 33, 39, 67]. Chart generation creates all syntactically sound sentences that express the meaning given grammar, vocabulary, and meaning. The active edge indicates elements that have not yet been formed, whereas inactive edges on charts indicate formed elements that are complete.

The process of writing poetry is an incremental one, involving multiple revisions, where each version aims to improve on the previous one until the poem is complete. Thus, it makes sense to use evolutionary algorithms for this purpose [26]. The main concept is to create an initial poem using a simple approach and evolve it over multiple generations in the direction of better poems. A fitness function determines a range of suitable characteristics for a poetic text. Applying crossover and mutation operations results in changes to the poem. Crossover integrates two existing poems to produce a new poem. This can be done by using the former syntax while retaining the current words or rhyme [26], or by exchanging some elements between the two [32]. The mutation is altering the rhyme, modifying a line's wording, or substituting keywords across the whole poem. Depending on the desired semantics, it can include editing, removing, or adding to the poem's content [32].

An alternative approach to this problem is to employ the Constraint Satisfaction technique by considering the number of limitations associated with the generation of poetry [51, 69]. The process of constraining satisfaction involves a solver that probes the search space to generate a solution. This solution fits the given attributes which are represented as the structure of a poem and dictionary terms. To produce diverse poems, different constraints might be imposed. For example, the soft constraints (e.g., rhymes) are optional, whereas the hard constraints (e.g., syllables in each line or numbers of lines) are mandatory.

Poetic writings have been produced by using language models, limited by the goal style and a predetermined form. It includes Markov models [4] and Recurrent Neural Networks (RNNs). The authors in [47] used the RNN to guess the immediate rap lyrics given a list of words. RNNs may also be used to progressively create new lines while keeping in mind the phonetics, structure, and semantics of the previous lines [76, 79]. One neural network (NN) may be used to choose line structure, while another to control the lines generated using individual words. Milanova et al. [35] tried to use different language generation models including RNN and LSTM. They also performed some experiments with transfer learning for poetry generation. Yolcu et al. [78]

trained an RNN model on a self-collected dataset in the Turkish language. Additionally, Yan et al. [76] made some advancements in the preceding procedures and introduced poetry polishing iterations. Ghazvininejad et al. [19] utilized a Finite-State Acceptor as an alternative for controlling rhyme and meter in the RNN language model. In [31], authors use the RankSVM with ANN for the completion of the next lines using poetic features like semantic similarity, structure, and rhyme from the song lyrics written by humans.

In [12], a poetry corpus was utilized for training Support Vector Machines (SVMs), which were used to predict the follow-up lines with certain syllables and rhyming characteristics. Furthermore, conventional NNs were also applied in [26] to evaluate the quality of poetry produced using an evolutionary method. In [9, 80], authors used a language translation model (Encoder-Decoder) for generating poetic data. In [73], the poetry is generated from prosaic text using Encoder-Decoder. Transfer learning has also been experimented with in different languages [38, 44, 57]. Saeed et al. [55] used Generative Adversarial Network (GAN) for poetry generation purposes.

In [65], authors use hierarchical RNNs. They use two separate models for generating the poetry. Bi-directional gated recurrent (Bi-GRU) units are used in the initial model to create the first line. To generate the next lines, they used an attention framework in a modified Bi-GRU encoder-decoder model. Their method is also divided into two parts. In the first part, they extract the keywords from some verses and then generate the poetry. In poetry generation, the context is maintained by considering all newly generated lines with keywords as input to generate the next line of the poem. Later, in Arabic, Beheitt et al. [5] used a GPT-2 for generating Arabic poems.

Automation of evaluating a piece of text as poetry is a very complex task due to several form and content features. Despite this subjective element that makes poetry evaluation difficult, it is becoming more and more popular to investigate various approaches for evaluating both the produced output and the production process. The system-generated poetry is usually evaluated by human experts according to poetic rules and structure. For example, the qualities of grammaticality, poeticness, and meaningfulness can be verified to some extent by the techniques used in [5, 32, 36]. However, they can also be evaluated by looking at the outcomes. In light of this, some researchers evaluated the output of their method using human evaluations of a collection of generated poems and scored the properties mentioned above. Other writers, who also considered human judgment to evaluate the quality of their findings, used questionnaires that evaluated different features with some overlap. Toivanen et al. [70] include emotions, quality of language, understandability, mental image, and liking. Rashel and Manurung [51] use structure, grammar, message, dictation, and expressiveness as evaluation.

In [1, 37, 51, 70], authors evaluated the system-generated poetry using a Turing test in which the results were matched with human-written poetry. The validity of the Turing test to evaluate computational creativity techniques opens a huge discussion. [46]. The main contention was how it places too much emphasis on the final output and not enough on the creative process. In [66], the author used the Turing Test to judge the system-generated poetry. It promotes the use of more straightforward methods, some of which may only aim to deceive a human judge into believing that a human has generated their results. In [11, 36], the FACE descriptive model [10] was also used to evaluate the poetry generation systems.

Few methods have attempted to process and analyze the poetry-generating systems or their outputs, and those have focused on less subjective aspects of poetry. For example, [35, 77] used ROUGE to evaluate the system-generated poetry, and BLEU was used by [5, 76, 79] to evaluate the generation of good verses. Goncalo Oliveira et al. [42] also used ROUGE to evaluate the variation of the system-generated poetry. In [75], the author used the average cosine similarity to find the semantic coherence in human and system-generated haikus. Potash et al. [47] also calculate the cosine similarity between original text and system-generated text. Malmi et al. [31] compute the rhyme density, which was also computed later by Goncalo Oliveira et al. [42]. Chen et al. [9] calculated the bigram-based Jaccard similarity to assess the generated poetry.

The distinctive contributions of this research are as follows.

(1) A poetry dataset is compiled focusing on a specific meter, consisting of over 20, 000 couplets from over 1, 000 poets.

(2) A meter-specific poetry generation system is proposed, using two different models for first and second verse generation. We used the LSTM model for the first verse and a sequence-to-sequence language translation model for the second verse generation.

(3) For evaluating our proposed approach, BELU score and human evaluation are employed to assess the deep learning models.

3 Data and Experiment

3.1 Data Collection

Urdu, being a low-resource language, presents challenges in finding poetry datasets, particularly in specific meters. Currently, there is a lack of high-quality datasets in this regard. Therefore, we collect our own meter-specific dataset of poetry. The summary of the dataset is given in Table 1. Rekhta [52] is one of the largest poetry data websites for Urdu and Hindi literature. We manually scrapped the data from the website of Rekhta. Our collected data contains 2787 poems, including 20911 couplets written by 1119 poets. Later, we removed 1806 couplets containing the poet's pen name (called مقطع). Now, we have 19105 simple verses. All these couplets are in one specific meter called behr-e-khafeef (خفیف بحرِ) with weight feet فاعلاتن مفاعلن فعلن. In poetry, both verses of a couplet convey a message, and most of the time, the position of the verse does not affect the meaning of the couplet. We decided to extend our dataset by changing the verse positions, and now we have 38210 couplets in our dataset.

3.1.1 Ethical Considerations. The data for this study was collected through manual scraping from Rekhta [52], a public website dedicated to Urdu and Hindi literature. Rekhta aggregates poetry contributed by poets, often with explicit permission for inclusion. While we have not formally requested permission to use the entire dataset for this research, it is important to note that Rekhta's collection is publicly available and intended to serve the poetry community.

Table 1 Dataset Details

	Verses	Tokens	Types
Simple Verses	19105	237158	10675
Verses with Poet Name	1806	22393	3630
Total Verses	20911	259551	11239

3.2 Data Preprocessing

We did the following normalization steps on the collected data:

(1) In Unicode encoding, some Urdu letters can be represented in single and double characters. For example, the letter آ can be written either joining ا (Unicode value: 106*F*) and ¯ (Unicode value: 3065) or in a single letter آ (Unicode value: 2062). Same applied for أ ،ؤ ،ئ ـُ etc. We replaced all occurrences of these double-letter characters with the respective single-letter characters.

(2) After the analysis of the dataset, we found some unnecessary characters like white spaces (UTF-8 characters \xe2\x80\x8b, \xe2\x80\x8c, \xef\xbb\xbf, etc.). All white spaces and tabs were removed.

(3) Punctuations and diacritic signs of the Urdu language were also removed to simplify the dataset.

Table 2 Primary and secondary Urdu characters set (present in dataset)

Primary	خ	ح	چ	ج	ٹ	ت	پ	ب	ا
	ص	ش	س	ژ	ز	ڑ	ر	ذ	ڈ	د
	ل	گ	ک	ق	ف	غ	ع	ظ	ط	ض
		ے	ی	ء	ہ	و	ن	م
Secondary	ۓ	ۃ	ة	ۂ	ھ	ں	ئ	ؤ	أ	آ

(4) To identify the starting and ending of the verses, we added three tags: "<s>" at the start of each couplet, "<m>" at the end of the first verse, and "<e>" at the end of each couplet.

There were 70 unique characters before normalization. After these normalization steps, there were 48 characters left, including space and new line characters. Five tag characters were added. Now dataset contains 53 characters, in which the Urdu language letters are 46 (including 9 secondary and 37 primary characters). All Urdu characters are listed in the table 2 below:

In table 3, some data examples are shown from the collected dataset after the normalization steps.

Table 3 Sample dataset

<div dir="rtl">

<s> تجھ سے وابستگی رہے گی ابھی <m> دل کو یہ بیکلی رہے گی ابھی <e>

<s> عیش نایائدار پر نازاں <m> آدمی کتنا بھولا بھالا ہے <e>

<s> کتنے دلچسپ لگنے لگنے ہیں <m> میرے قصے تری زبانی بھی <e>

<s> دل میں ہر وقت یاس رہتی ہے <m> اب طبیعت اداس رہتی ہے <e>

<s> منفعت باٹتا رہا کل تک <m> آج بے سود ہو گا ہوں میں <e>

<s> کچھ تو دوست تھے بھی وفا دشمن <m> کچھ مری آنکھ میں حیا بھی تھی <e>

<s> خود سوال آپ ہی جواب ہوں میں <m> زندگی کی کھلی کتاب ہوں میں <e>

</div>

3.3 Methdology

Our proposed method comprises two main techniques, i.e., (i) first verse generation and (ii) second verse generation. The block diagram of the overall method is shown in Figure 1. In the end, both the first and second verses are concatenated to form a couplet. To generate the first verse, the character-level LSTM model is used, and to generate the second verse, the Encoder-decoder model is used for machine translation.

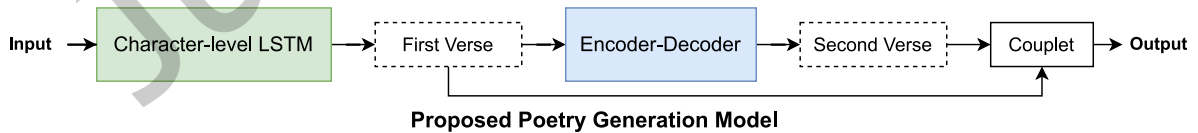**Proposed Poetry Generation Model**

Fig. 1 Block diagram of the proposed model

For the first verse generation, we used a character-based LSTM model with the collected dataset on the character level. The block diagram of the character-based LSTM is shown in Figure 2. It contains an embedding layer, which is used to convert a character into a feature vector, three hidden layers with the same number of
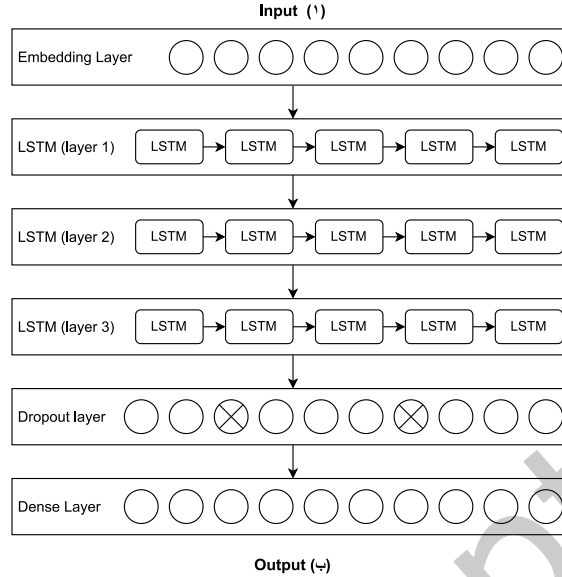
Fig. 2 First verse generation using character-level LSTM

neurons in each layer, and a dropout layer for regularization and to reduce overfitting. In the training step, the loss is reduced in the backpropagation step which follows the standard backpropagation method of LSTM. This model generates the first verse of the couplet. The first verse will be generated at the output with the variable length.

Once the LSTM model generates the first verse of the couplet, we give it as input to the encoder-decoder model. The encoder-decoder model works as a sequence-to-sequence model by passing the first verse to the encoder for the generation of the context vector. The encoder-decoder model is composed of GRU cells for paying attention to the meter. The attention weights can be set constant or in decreasing order as the sequence of words in a verse increases. This context vector is then passed to the attention unit and to the decoder for the generation of the second verse.

Once the LSTM model generates the first verse of the couplet, we give it as input to the encoder-decoder model shown in Figure 3. The encoder-decoder model works as a sequence-to-sequence model by passing the first verse to the encoder for the generation of the context vector. The encoder-decoder model is composed of GRU cells for paying attention to the meter. The attention weights can be set constant or in decreasing order as the sequence of words in a verse increases. This context vector is then passed to the attention unit and to the decoder for the generation of the second verse.

3.4 Evaluation Matrix — BLEU

BLEU (Bilingual Evaluation Understudy) is a metric for evaluating machine-translation output quality. It was developed by Kishore Papineni et al. [45] in 2002 and has since become one of the most widely used metrics for machine translation. There are several variations of the BLEU metric, including BLEU-1, BLEU-2, BLEU-3, and BLEU-4. BLEU-1 is the simplest version of BLEU, which compares the machine translation output to the reference translation using only unigrams (individual words). Similarly, the other BLEU scores like BLEU-2, BLEU-3,

context vector

attention vector

attention weights

0.4 0.3 0.1 0.1 0.1

قصهٔ غم لب آشنا کرتے </s>

عمر گزری ہے التجا کرتے <s> قصهٔ غم لب آشنا کرتے
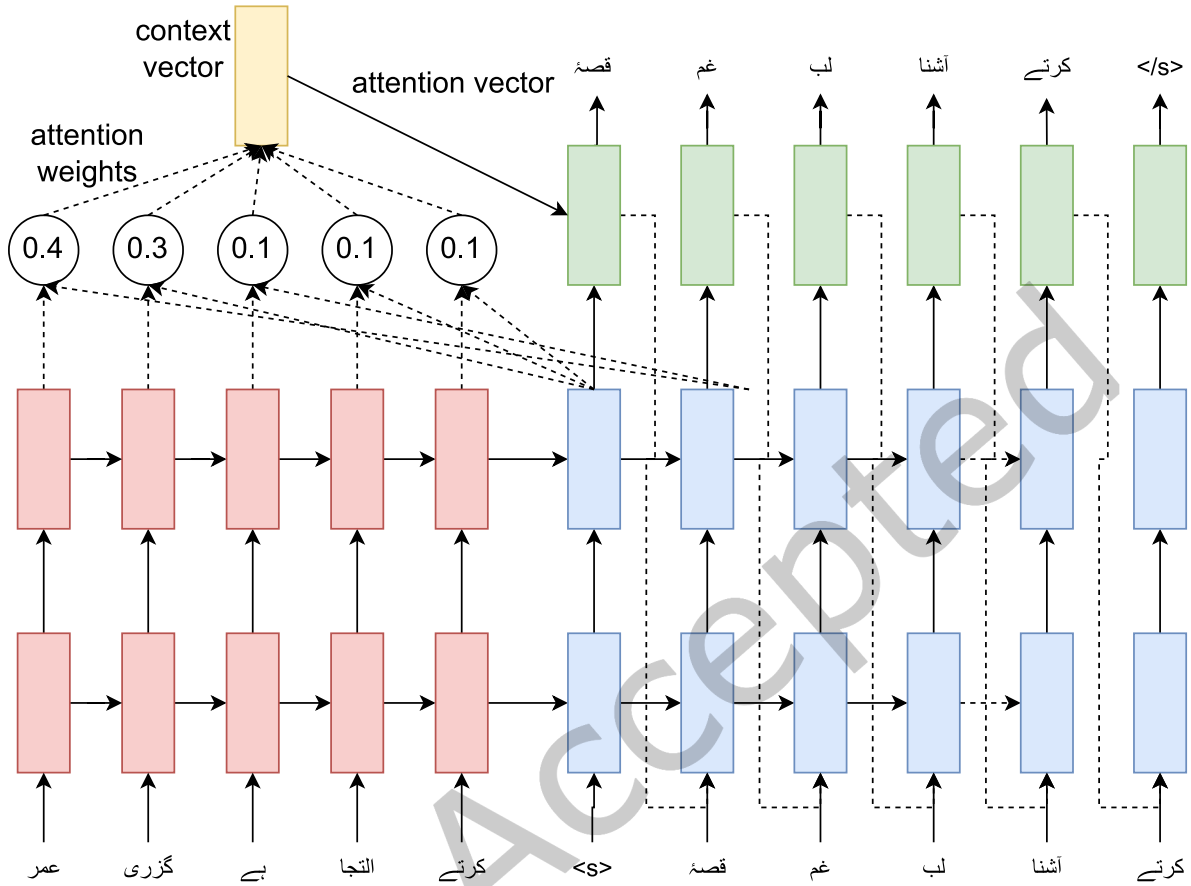
Fig. 3 Encoder decoder model for the second verse generation.

and BLEU-4 compare machine translation output to the reference translation using bigrams (pairs of consecutive words), trigrams (triplets of consecutive words), and quadrigrams (four consecutive words). The BLEU scores are computed as the geometric mean of the precision of the n-gram with a penalty for short translations.

The 1 is the formula for computing the BLEU score:

$$BLEU = BP \cdot \exp\left(\sum_{i=0}^{n} w_n \log\left(p_n\right)\right) \tag{1}$$

where BP is the brevity penalty defined in 2, which is a factor that penalizes short translations, w_n is the weight between 0 and 1 for $\log p_n$, and p_n is the precision of the n-grams in the machine translation output, which is defined as the number of n-grams in the machine translation output that also appear in the reference translation, divided by the total number of n-grams in the machine-translation output.

$$BP = \begin{cases} 1 & if \, c > 0 \\ \exp\left(1 - \frac{r}{c}\right) & if \, c \leq 0 \end{cases} \tag{2}$$

where c is the total number of n-grams in all candidate lines and r is the matching lengths for each line.

4 Experiments and Results

4.1 Experimental Setup

We used two different models to generate each verse of the couplet. Firstly, we trained the LSTM model on character level due to a smaller and limited dataset. we trained the LSTM model on our collected dataset for 100 epochs with a batch size of 32 with a learning rate of 0.001 using the Adam optimizer. We introduced a *randomness* variable to control the randomness in the generated output. We used three layers of LSTM with 512 units in each layer. We used a dropout layer and predicted the next character until the verse was completed. We concatenated the characters to form the complete verse and used it as input in the next model. For the second verse generation, we used the language translation models. We used the Encoder-Decoder model to complete the couplet by predicting the next verse. We used 1024 units of GRU in both the encoder and decoder layers. We trained this model for 50 epochs with a batch size of 64 and with a learning rate of 0.001. We used the BLEU score to evaluate the performance of this model. We also used Bahdanau's attention in this model.

Once our proposed model (UPON) is trained, we use the LSTM model to generate the first verse by providing a seed input. The Aruuz API [2] plays a crucial role in evaluating the poeticness of the generated verses, specifically by verifying adherence to behr-e-khafeef meter. This API is essential because meter adherence is a defining feature of Urdu poetry, impacting both structure and rhythm. Known for its reliable performance in meter detection, the Aruuz API analyzes each verse's syllabic structure against the designated meter, ensuring that the generated verse matches traditional poetic standards. We passed the generated verse with behr-e-khafeef meter feet and verified if the model generated the verse in the specific meter or not. If the generated verse is not in the meter, we generate the verse again.

For generating the second verse, we employ a sequence-to-sequence (seq-to-seq) model, which operates similarly to a machine translation task. Here, the second verse is generated by treating the first verse as a seed and passing it through the seq-to-seq model. This process helps ensure that the second verse is contextually relevant to the first one, thus creating a more coherent couplet. However, we observed that the seq-to-seq model generated repetitive output when trained solely for second verse generation. To overcome this issue, we integrated the LSTM model again for generating more diverse second verses. The LSTM model, when fed with the first verse as input, generates varied and fluently constructed second verses that align with the desired poetic structure. This two-model approach allows us to combine the strengths of both models: the seq-to-seq model's ability to generate contextually linked second verses, and the LSTM's flexibility in creating diverse outputs.

Furthermore, if the first verse is flawed or lacks coherence, the system allows re-generation of the first verse, providing flexibility and ensuring that the final couplet maintains both structural and thematic integrity. This approach is also supported by literature in poetry generation, where distinct models are often used for generating different parts of a poem (such as the first and second verses) to ensure coherence, thematic alignment, and variety. By separating the tasks and leveraging the strengths of each model, we believe this approach improves the quality and diversity of generated poetry compared to using a single model for both verses.

4.2 Experimental Results and Limitations

4.2.1 Experimental Results for First Verse Generation Model. In the LSTM model, we trained the model with 100 epochs. The learning curves are shown in Figure 4. The training accuracy is shown in Figure 4a, and the training
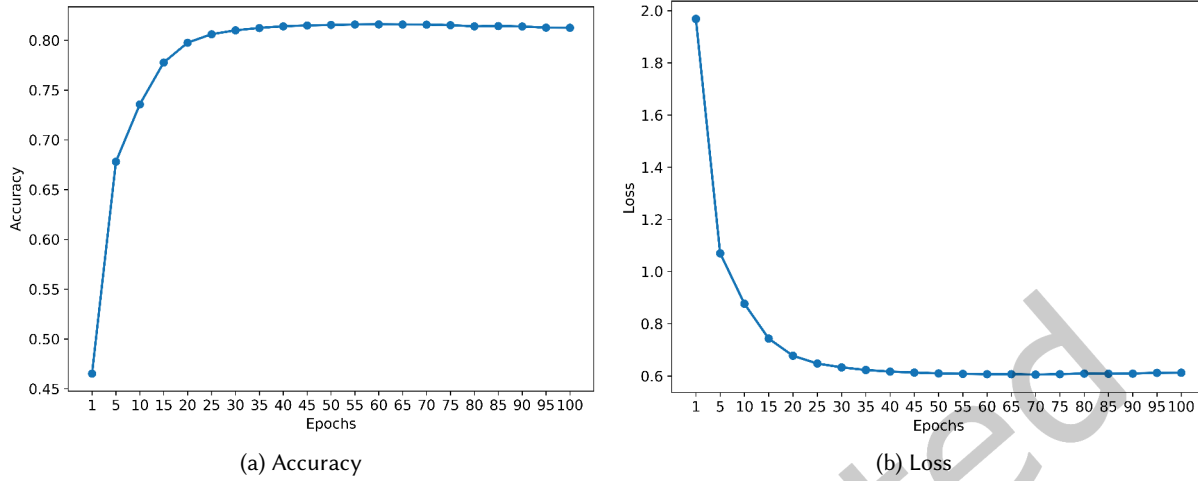
(a) Accuracy

(b) Loss

Fig. 4 Learning curves of the LSTM model (a) accuracy and (b) loss.

Table 4 First verse generation by LSTM model

Input seed	\<s\>	اجنبی	لگی دل
Generated Verses	طرح کی خزاں گئی آ کر ٹوٹ ۔۔ کو دل دیا کر شیشہ اجنبی	قدر و رقیب کیجئے لگی دل
	تھا میں ہوں یہاں سے مدتوں ۔۔ ابھی ہے سنی کی شام اجنبی	ہے شکاری سوا کے لگی دل
	جائے آزباں ہے اپنی گفتگو ۔۔ ڈھونڈیں چلو میں شہر اجنبی	ہے دیا دکھائی تو لگی دل
	تھا گھر مرا نے فریاد میری ۔۔ پر زمینوں پر شہر اجنبی	ادھر ہے چاھتی کی ان لگی دل
	ایسا ہے قرار ہے اگر دل ۔۔ جاؤں پلٹ میں شہر اجنبی	جاتی نہیں سبب ہے لگی دل

loss across the number of epochs is shown in Figure 4b. From the figures, it can be seen that as the number of epochs rises, accuracy increases while loss decreases.

We also introduced a meter confidence score that shows the accuracy rate of the LSTM model. We generate the verses multiple times and evaluate their meter with Aruuz API [2]. We divide the correct meter verses with the total verses. The formula for calculating the meter confidence score is as follows:

$$Meter\ Confidence = \frac{Verses\ With\ Correct\ Meter}{All\ Verses} \tag{3}$$

We calculate the meter confidence with this equation 3 and achieve the score of 69.44%.

On a given input seed, the LSTM model generates a new verse each time. We tried different input seeds and generated 5 verses for every input seed. The table 4 shows a sample of generated verses on different inputs.

4.2.2 Experimental Results for Second Verse Generation Model. We trained the encoder-decoder model for the second verse generation with 50 epochs. Figure 5 shows the learning curves. The BLEU-4 score is shown in Figure 5a, and the training loss across the number of epochs is shown in Figure 5b.

On giving an input verse, the encoder-decoder model generates the second verse. We tried different inputs and Table 5 shows the outputs.

Table 5 Second verse generation by encoder-decoder

First Verse (Input)	Second Verse (Output)
مجھے ہے ڈھونڈھتا تو میں مفلسی	ہے حالت میری بیدار تو اب
نہیں تو گم میں سہی ھی مختصر	نہیں بھی میں کائنات کی غیر
پر در ترے ہے بعد مدتوں	ہیں اٹھائے سر سے آنکھوں میری
ہے سمجھتا زندگی کو موت	کو مجھ ہے داستان کی درد
ہے چلتی ساتھ میرے کر ٹوٹ	سے آنکھوں ہے زندگی بھر رات

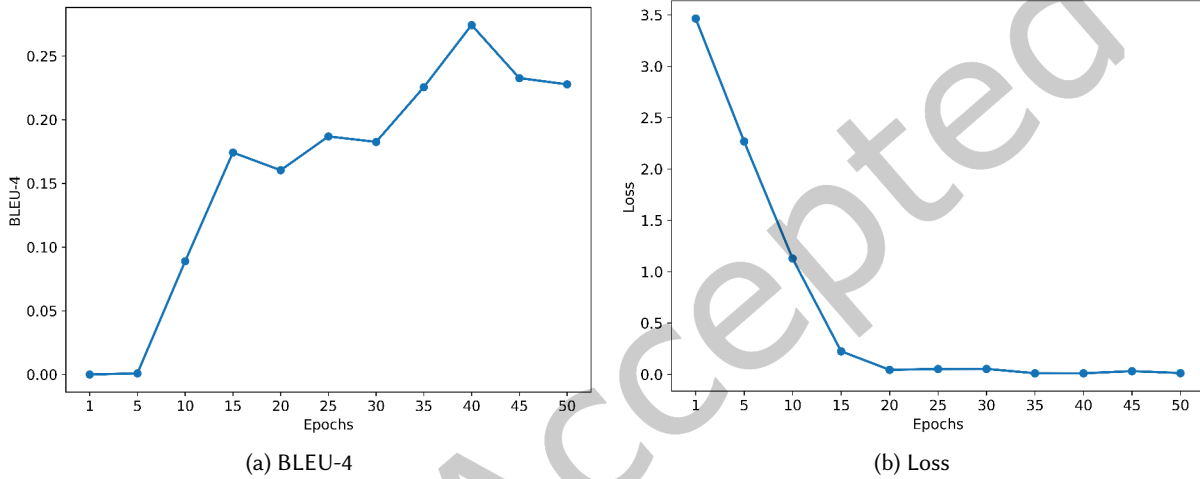(a) BLEU-4   (b) Loss

Fig. 5 Learning curves of the encoder-decoder model (a) BLEU-4 and (b) loss.

4.3 Comparison with Existing Models

We compare our approach with existing models such as Vanilla, LSTM, GRU, RNN, Bi-GRU, and GPT-2 and achieve the best scores. Table 6 shows the BLEU score comparison between the existing work done on poetry generation and our proposed model (UPON) on our collected dataset. We can see that we achieve the BLEU-4 score of 0.2763 which is the best score so far.

4.4 Human Evaluation

Evaluation of poetry data is more complex than text data due to its poetic features. We cannot evaluate it by using text evaluation methods. We have defined human evaluation matrices in the *Introduction* Section used in many poetry generation research. We evaluate the generated poetry from four poets and Table 7 shows the results. After getting results from the poets, we compare our results with the state-of-the-art models in Table 8. Since the collected dataset is in one specific meter, we achieve a poeticness score of 4.06 out of 5 which is the best score so far. We achieve a fluency score of 2.03 out of 5 which is comparable with other poetry generation models. However, we get low coherence and meaning scores of 1.01 out of 5 and 1.05 out of 5 respectively. This is because of a limited and generic dataset. These results could get better with a larger dataset.

Table 6 BLEU scores comparison of the UPON with existing methods

Models	BLEU Evaluation			
	BLEU-4	BLEU-3	BLEU-2	BLEU-1
Vanilla	0	0	0.0199	0.0211
LSTM	0.0013	0.0081	0.1124	0.1522
GRU	0.0021	0.0084	0.1139	0.1512
RNN Encoder-Decoder (without attention)	0.0510	0.0740	0.1539	0.2513
RNN Encoder-Decoder (attention)	0.0801	0.0911	0.2110	0.3010
Bi-GRU Encoder-Decoder (attention)	0.1092	0.2040	0.3144	0.4122
GPT-2 Model + only pre-training	0	0.0395	0.1737	0.5535
GPT-2 Model + pre-training + fine-tuning	0.1871	0.3230	0.5369	0.8739
UPON: Urdu Poetry Generation	**0.2763**	**0.3845**	**0.4375**	**0.5191**

4.4.1 Instructions for Human Evaluators. Human evaluators were selected based on their familiarity with Urdu poetry and their ability to analyze poetic quality. Each evaluator was instructed to review the couplets based on four primary criteria: fluency, poeticness, coherence, and meaning. Evaluators were informed that:

- **Fluency**: This refers to the natural flow and readability of the couplet, ensuring that it sounds like authentic Urdu poetry without awkward phrasing or structural errors.
- **Poeticness**: It refers to the aesthetic quality and adherence to poetic structure, including meter, rhyme, and refrain. Evaluators were asked to assess whether the couplet had the rhythmic and stylistic elements typical of traditional Urdu poetry.
- **Coherence**: This considers how logically connected the two lines of the couplet are and if they form a unified message. Evaluators were to identify if each verse pair appeared as a single, coherent idea.
- **Meaning**: It reflects the depth and interpretive quality of the couplet. Evaluators rated whether the couplet carried a meaningful or thoughtful message, beyond simple or surface-level text.

Each evaluator assigned a score from 1 to 5 for each criterion, following the rubric below:

(1) – **Very Poor**: The couplet lacks fluency, poetic quality, coherence, or meaningful content entirely.
(2) – **Poor**: Some poetic elements may be present, but the couplet generally lacks fluency, coherence, or meaningful depth.
(3) – **Average**: The couplet meets basic poetic structure with limited coherence and meaning, but may lack fluency or depth.
(4) – **Good**: The couplet generally exhibits fluency and coherence, meets poetic standards, and conveys some meaningful content.
(5) – **Excellent**: The couplet fully adheres to poetic form, is fluent and coherent, and presents a deep or impactful message.

Evaluators rated each couplet independently without interaction to ensure unbiased assessment. Scores were averaged for each criterion to measure the overall quality of the generated poetry.

4.5 Limitations

Despite achieving a good poeticness score (4.06 out of 5) for a poetry generation model on a dataset of Urdu poetry, there are certain limitations in the current approach. One significant challenge is the low coherence and meaning scores (1.01 and 1.05 out of 5) in the human evaluation. These results reflect the difficulty of generating semantically coherent poetry.

The primary reason for these low scores is the limited and generic dataset used for training the model. With only 20,000 couplets in one specific meter (behr-e-khafeef), the model faces constraints in learning deeper semantic relationships and the complexities of different poetic forms. While the model performs well in terms of fluency and poeticness, the dataset size and meter specificity hinder its ability to generate more coherent and meaningful poetry. We acknowledge that the dataset size is a limiting factor. Expanding it to include a wider variety of meters and poetic forms would enhance the model's capacity to generate poetry with better meaning and coherence. Furthermore, improving the training methodology and incorporating larger, more diverse data could help address this limitation in future iterations of the model.

The focus on a single meter was initially chosen because this meter is more common in traditional Urdu poetry. The focus on this meter allowed the model to be trained effectively while addressing a gap in existing research. However, this limits the model's applicability to other meters or free verse poetry, which would involve varying rhythmic and structural patterns. Future work could aim to expand the dataset to include a variety of meters, as well as free verse, allowing for a more comprehensive evaluation of the model's capabilities and improving its versatility.

Another limitation is the BLEU-4 score of 0.2763, the highest reported in Urdu poetry generation but still relatively low compared to other domains. This score primarily reflects n-gram overlap, which may not fully capture poetic qualities such as meaning, creativity, and emotion. The limited dataset and focus on a single meter likely contribute to this modest BLEU score. Expanding the dataset to include various meters and styles could improve both fluency and BLEU scores. While the current score indicates room for improvement, it also highlights the challenges of poetry generation in low-resource languages. Future improvements, such as larger datasets, data augmentation, and advanced training techniques, are expected to enhance performance.

Table 7 Comparison of human evaluation scores of the UPON model with existing models

Models	Human Evaluation (by poets)			
	Poeticness	Fluency	Meaning	Coherence
Vanilla	0	0.1	0.7	0.8
LSTM	0.1	0.3	0.8	0.9
GRU	0.2	0.3	1	1
RNN Encode-Decoder (without attention)	0.3	2	2.4	1.5
RNN Encode-Decoder (attention)	0.4	2.3	2.7	2.5
Bi-GRU Encode-Decoder (attention)	0.4	2.1	2.7	3.2
GPT-2 Model + only pre-training	0	1.5	1.5	1.3
GPT-2 Model + pre-training + fine-tuning	3.4	2.8	2.6	2.6
UPON: Urdu Poetry Generation	**4.06**	**2.03**	**1.05**	**1.01**

5 Acknowledgement

Table 8 Human evaluation scores from poets (F: Fluency, C: Coherence, M: Meaning, P: Poeticness)

Verses	Poet 1				Poet 2				Poet 3				Poet 4			
	F	C	M	P	F	C	M	P	F	C	M	P	F	C	M	P
ھے آتی سے کہاں خموشی کب آتی نہیں ہو تو ھے تو آپ	0	0	0	5	2	0	0	5	1	0	0	5	2	1	1	5
میں ہوں گا ڈر کا پتھر کو سب میں ہوں گا میں سے خود جگہ اس	0	0	0	5	0	0	0	5	3	0	0	5	2	1	2	5
میں پانی اداس تھا قدر اس تھا سمندر میں خواب اک جو میں	3	3	0	5	5	2	2	5	3	3	2	5	4	4	4	5
لیکن لوں چوم کو آنکھوں نیند ھے کم حوصلہ کا دل کروں گا	0	2	0	5	5	2	2	5	3	4	4	5	5	5	5	5
مطلب کیا سے جہاں اس وہ جیسے ہیں رہے کو احترام یہ تو یہ	0	0	0	1	3	0	0	5	3	0	2	5	2	0	2	4
ہیں کرتے تمام زمانہ اک میں دنیا کے اس ہیں الہی یا	2	1	0	5	2	0	2	5	2	0	0	5	2	2	2	5
ھے عالم پہ اس میں دنیا ایک کچھ سے آدمی تصویر ایک	1	1	1	5	0	0	0	5	3	0	2	5	2	3	1	5
کرنا نہیں خطا میری میں اس کرتا نہیں مجھے بس مرتکب	3	0	0	5	0	0	3	5	5	0	0	5	4	3	4	5
جاناں قافلے کے مجنوں تو یہ گا دے حوصلے اپنے خرد یہ	3	0	2	5	3	0	0	5	1	3	2	5	3	3	2	5
جاؤ گزر سے رنگ ہر جلوہ جائے ہو جسم دور قدم ہر	5	3	3	5	0	0	0	2	3	0	0	5	5	3	3	5
پہنچی تک آسمان نہیں کچھ لیا دیکھ کے سن ہیں ٹکڑے کے سب	2	0	0	5	2	3	2	5	5	1	2	5	3	3	3	5
گے دیکھیں بار لاکھ آسماں گے دیکھیں پھر چاند ہی چاند	4	3	3	5	2	2	2	2	0	1	0	0	3	3	4	4

6 Conclusion

In this paper, we have collected a dataset of meter-specific poetry in a low-resource language Urdu. To the best of our knowledge, it is the first meter-specific dataset that includes more than 20000 couplets from more than 1000 poets. We also proposed an Urdu poetry generation model using deep-learning methods to produce both verses of a couplet. We used LSTM to produce the first verse and an Encoder-Decoder for the second verse to complete the couplet. We have used the accuracy and BLEU scores to examine the learning of the LSTM and Encoder-Decoder models respectively. Our model achieves the BLEU-4 score of 0.2763 which is comparable to the other state-of-the-art models like Vanilla, LSTM, GRU, Bi-GRU, etc. We defined the meter confidence score as the accuracy of generating correct meter verses. We calculate the meter confidence of our model and we achieve a score of 69.44%. We also assess the output of our proposed model on grounds of fluency, coherence, meaning, and poeticness by performing the human evaluation from 4 poets. We achieved a poeticness score of 4.2 out of 5 which is the best poeticness score so far. We achieve a fluency score of 2.03 out of 5 which is comparable to other state-of-the-art poetry generation models.

References

[1] Manex Agirrezabal, Bertol Arrieta, Aitzol Astigarraga, and Mans Hulden. 2013. POS-tag based poetry generation with WordNet. In *Proceedings of the 14th European Workshop on Natural Language Generation*. 162–166.

[2] Syed ZeeShan Asghar. 2023-02-02. تقطیع کی اشعار. (2023-02-02). https://www.aruuz.com/taqti Accessed on February 17, 2023.

[3] Abu Bakar, Raheem Sarwar, Saeed-Ul Hassan, and Raheel Nawaz. 2023. Extracting Algorithmic Complexity in Scientific Literature for Advance Searching. *Journal of Computational and Applied Linguistics* 1 (2023), 39–65.

[4] Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. 2012. Markov Constraints for Generating Lyrics with Style.. In *Ecai*, Vol. 242. 115–120.

[5] Mohamed El Ghaly Beheitt and Moez Ben Haj Hmida. 2022. Automatic Arabic Poem Generation with GPT-2.. In *ICAART (2)*. 366–374.

[6] Camille Bloomfield and Hélène Campaignolle-Catel. 2016. Machines littéraires, machines numériques: l'Oulipo et l'informatique.

[7] Laurent Bourbeau, Denis Carcagno, Eli Goldberg, Richard Kittredge, and Alain Polguere. 1990. Bilingual generation of weather forecasts in an operations environment. In *COLING 1990 Volume 3: Papers presented to the 13th International Conference on Computational Linguistics*.

[8] Selmer Bringsjord and David Ferrucci. 1999. *Artificial intelligence and literary creativity: Inside the mind of brutus, a storytelling machine.* Psychology Press.

[9] Huimin Chen, Xiaoyuan Yi, Maosong Sun, Wenhao Li, Cheng Yang, and Zhipeng Guo. 2019. Sentiment-Controllable Chinese Poetry Generation.. In *IJCAI*. 4925–4931.

[10] Simon Colton, John William Charnley, and Alison Pease. 2011. Computational Creativity Theory: The FACE and IDEA Descriptive Models.. In *ICCC*. Mexico City, 90–95.

[11] Simon Colton, Geraint A Wiggins, et al. 2012. Computational creativity: The final frontier?. In *Ecai*, Vol. 12. Montpelier, 21–26.

[12] Amitava Das and Björn Gambäck. 2014. Poetic Machine: Computational Creativity for Automatic Poetry Generation in Bengali.. In *ICCC*. 230–238.

[13] Sobha Lalitha Devi et al. 2010. An alternate approach towards meaningful lyric generation in tamil. In *Proceedings of the NAACL HLT 2010 Second Workshop on Computational Approaches to Linguistic Creativity*. 31–39.

[14] Belén Díaz-Agudo, Pablo Gervás, and Pedro A González-Calero. 2002. Poetry generation in COLIBRI. In *European Conference on Case-Based Reasoning*. Springer, 73–87.

[15] Rachel A. Fleming-May and Harriett E. Green. 2016. Digital innovations in poetry: Practices of creative writing faculty in online literary publishing. *J. Assoc. Inf. Sci. Technol.* 67, 4 (2016), 859–873.

[16] Pablo Gervás. 2000. Wasp: Evaluation of different strategies for the automatic generation of spanish verse. In *Proceedings of the AISB-00 symposium on creative & cultural aspects of AI*. 93–100.

[17] Pablo Gervás. 2001. An expert system for the composition of formal spanish poetry. In *Applications and innovations in intelligent systems VIII*. Springer, 19–32.

[18] Pablo Gervás, Birte Lönneker-Rodman, Jan Christoph Meister, and Federico Peinado. 2006. Narrative models: Narratology meets artificial intelligence. In *International Conference on Language Resources and Evaluation. Satellite Workshop: Toward Computational Models of Literary Analysis*. Génova, 44–51.

[19] Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1183–1191.

[20] Muhammad Umair Hassan, Saleh Alaliyat, Raheem Sarwar, Raheel Nawaz, and Ibrahim A Hameed. 2023. Leveraging deep learning and big data to enhance computing curriculum for industry-relevant skills: A Norwegian case study. *Heliyon* 9, 4 (2023).

[21] Muhammad Umair Hassan, Xiuyang Zhao, Raheem Sarwar, Naif R Aljohani, and Ibrahim A Hameed. 2023. SODRet: Instance retrieval using salient object detection for self-service shopping. *Machine Learning with Applications* (2023), 100523.

[22] Saeed-Ul Hassan, Naif Radi Aljohani, Usman Iqbal Tarar, Iqra Safder, Raheem Sarwar, Salem Alelyani, and Raheel Nawaz. 2023. Exploiting tweet sentiments in altmetrics large-scale data. *Journal of Information Science* 49, 5 (2023), 1229–1245.

[23] Hani D Hejazi, Ahmed A Khamees, Muhammad Alshurideh, and Said A Salloum. 2021. Arabic text generation: Deep learning for poetry synthesis. In *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2021*. Springer, 104–116.

[24] S Kim. 2002. Artequakt: Generating tailored biographies with automatically annotated fragments from the web, semantic authoring. In *Annotation and Knowledge Markup Workshop in the 15th European Conference on Artificial Intelligence, 2002*.

[25] Sankar Kuppan, Sobha Lalitha Devi, et al. 2009. Automatic generation of Tamil lyrics for melodies. In *Proceedings of the workshop on computational approaches to linguistic creativity*. 40–46.

[26] Robert P Levy. 2001. A computational model of poetic creativity with neural network as measure of adaptive fitness. In *Proceedings of the ICCBR-01 Workshop on Creative Systems*. Citeseer.

[27] Danielle Lewis, Andrea Zugarini, and Eduardo Alonso. 2021. Syllable neural language models for English poem generation. In *12th International Conference on Computational Creativity (ICCC'21)*.

[28] Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. 2020. Domain adaptation of Thai word segmentation models using stacked ensemble. Association for Computational Linguistics.

[29] Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. 2021. Handling cross and out-of-domain samples in Thai word segmentation. Association for Computational Linguistics.

[30] Yingfeng Luo, Changliang Li, Canan Huang, Chen Xu, Xin Zeng, Binghao Wei, Tong Xiao, and Jingbo Zhu. 2021. Chinese Poetry Generation with Metrical Constraints. In *Natural Language Processing and Chinese Computing: 10th CCF International Conference, NLPCC 2021, Qingdao, China, October 13–17, 2021, Proceedings, Part I 10*. Springer, 377–388.

[31] Eric Malmi, Pyry Takala, Hannu Toivonen, Tapani Raiko, and Aristides Gionis. 2016. Dopelearning: A computational approach to rap lyrics generation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 195–204.

[32] Hisar Manurung. 2004. An evolutionary algorithm approach to poetry generation. (2004).

[33] Hisar Maruli Manurung. 1999. A chart generator for rhythm patterned text. In *Proceedings of the First International Workshop on Literature in Cognition and Computer*. 15–19.

[34] Stephen McGregor, Matthew Purver, and Geraint Wiggins. 2016. Process based evaluation of computer generated poetry. In *Proceedings of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation*. 51–60.

[35] Ivona Milanova, Ksenija Sarvanoska, Viktor Srbinoski, and Hristijan Gjoreski. 2019. Automatic text generation in Macedonian using recurrent neural networks. In *ICT Innovations 2019. Big Data Processing and Mining: 11th International Conference, ICT Innovations 2019, Ohrid, North Macedonia, October 17–19, 2019, Proceedings 11*. Springer, 1–12.

[36] Joanna Misztal and Bipin Indurkhya. 2014. Poetry generation system with an emotional personality.. In *ICCC*. 72–81.

[37] Yael Netzer, David Gabay, Yoav Goldberg, and Michael Elhadad. 2009. Gaiku: Generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*. 32–39.

[38] Tuan Nguyen, Phong Nguyen, Hanh Pham, Truong Bui, Tan Nguyen, and Duc Luong. 2021. SP-GPT2: Semantics Improvement in Vietnamese Poetry Generation. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 1576–1581.

[39] Hugo Gonçalo Oliveira. 2012. PoeTryMe: a versatile platform for poetry generation. *Computational Creativity, Concept Invention, and General Intelligence* 1 (2012), 21.

[40] Hugo Gonçalo Oliveira. 2015. Tra-la-lyrics 2.0: Automatic generation of song lyrics on a semantic domain. *Journal of Artificial General Intelligence* 6, 1 (2015), 87.

[41] Hugo Gonçalo Oliveira and Ana Oliveira Alves. 2016. Poetry from concept maps–yet another adaptation of PoeTryMe's flexible architecture. In *Proceedings of 7th International Conference on Computational Creativity, ICCC*.

[42] Hugo Gonçalo Oliveira, Raquel Hervás, Alberto Díaz, and Pablo Gervás. 2017. Multilingual extension and evaluation of a poetry generator. *Natural Language Engineering* 23, 6 (2017), 929–967.

[43] Hugo R Gonçalo Oliveira, F Amilcar Cardoso, and Francisco C Pereira. 2007. Tra-la-Lyrics: An approach to generate text based on rhythm. In *Proceedings of the 4th. International Joint Workshop on Computational Creativity*. A. Cardoso and G. Wiggins.

[44] Aitor Ormazabal, Mikel Artetxe, Manex Agirrezabal, Aitor Soroa, and Eneko Agirre. 2022. Poelm: A meter-and rhyme-controllable language model for unsupervised poetry generation. *arXiv preprint arXiv:2205.12206* (2022).

[45] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 311–318. https://doi.org/10.3115/1073083.1073135

[46] Alison Pease and Simon Colton. 2011. On impact and evaluation in computational creativity: A discussion of the Turing test and an alternative proposal. In *Proceedings of the AISB symposium on AI and Philosophy*, Vol. 39. Citeseer.

[47] Peter Potash, Alexey Romanov, and Anna Rumshisky. 2015. Ghostwriter: Using an lstm for automatic rap lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1919–1924.

[48] Álvaro Pérez Pozo, Javier de la Rosa, Salvador Ros, Elena González-Blanco, Laura Hernández, and Mirella De Sisto. 2022. A bridge too far for artificial intelligence?: Automatic classification of stanzas in Spanish poetry. *J. Assoc. Inf. Sci. Technol.* 73, 2 (2022), 258–267.

[49] Oxford University Press. 2023-02-02. Poetry. *Home: Oxford English Dictionary* (2023-02-02). https://www.oed.com/oed2/00182463 Accessed on February 17, 2023.

[50] Raymond Queneau. 1961. 100.000. 000.000. 000 de poemes. *Gallimard Series. Schoenhof's Foreign Books, Incorporated* (1961).

[51] Fam Rashel and Ruli Manurung. 2014. Pemuisi: a constraint satisfaction-based generator of topical Indonesian poetry.. In *ICCC*. 82–90.

[52] Rekhta Foundation. [n. d.]. Urdu poetry, Urdu Shayari of famous poets. *Rekhta* ([n. d.]). http://www.rekhta.org/ Accessed on February 17, 2023.

[53] Fahad Sabah, Yuwen Chen, Zhen Yang, Muhammad Azam, Nadeem Ahmad, and Raheem Sarwar. 2023. Model optimization techniques in personalized federated learning: A survey. *Expert Systems with Applications* (2023), 122874.

[54] Fahad Sabah, Yuwen Chen, Zhen Yang, Abdul Raheem, Muhammad Azam, and Raheem Sarwar. 2023. Heart Disease Prediction with 100% Accuracy, Using Machine Learning: Performance Improvement with Features Selection and Sampling. In *2023 8th IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC)*. IEEE, 41–45.

[55] Asir Saeed, Suzana Ilić, and Eva Zangerle. 2019. Creative GANs for generating poems, lyrics, and metaphors. *arXiv preprint arXiv:1909.09534* (2019).

[56] Iqra Safder, Hafsa Batool, Raheem Sarwar, Farooq Zaman, Naif Radi Aljohani, Raheel Nawaz, Mohamed Gaber, and Saeed-Ul Hassan. 2022. Parsing AUC result-figures in machine learning specific scholarly documents for semantically-enriched summarization. *Applied*

Artificial Intelligence 36, 1 (2022), 2004347.

[57] Marvin C Santillan and Arnulfo P Azcarraga. 2020. Poem generation using transformers and Doc2Vec embeddings. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–7.

[58] Raheem Sarwar. 2022. Author Gender Identification for Urdu Articles. In *International Conference on Computational and Corpus-Based Phraseology*. Springer, 221–235.

[59] Raheem Sarwar and Saeed-Ul Hassan. 2021. Urduai: Writeprints for Urdu authorship identification. *Transactions on Asian and Low-Resource Language Information Processing* 21, 2 (2021), 1–18.

[60] Raheem Sarwar, Thanasarn Porthaveepong, Attapol Rutherford, Thanawin Rakthanmanon, and Sarana Nutanong. 2020. StyloThai: A scalable framework for stylometric authorship identification of thai documents. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 19, 3 (2020), 1–15.

[61] Raheem Sarwar, Attapol T Rutherford, Saeed-Ul Hassan, Thanawin Rakthanmanon, and Sarana Nutanong. 2020. Native language identification of fluent and advanced non-native writers. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 19, 4 (2020), 1–19.

[62] Raheem Sarwar, Norawit Urailertprasert, Nattapol Vannaboot, Chenyun Yu, Thanawin Rakthanmanon, Ekapol Chuangsuwanich, and Sarana Nutanong. 2020. *CAG*: Stylometric authorship attribution of multi-author documents using a co-authorship graph. *IEEE Access* 8 (2020), 18374–18393.

[63] Raheem Sarwar, Chenyun Yu, Ninad Tungare, Kanatip Chitavisutthivong, Sukrit Sriratanawilai, Yaohai Xu, Dickson Chow, Thanawin Rakthanmanon, and Sarana Nutanong. 2018. An effective and scalable framework for authorship attribution query processing. *IEEE Access* 6 (2018), 50030–50048.

[64] Raheem Sarwar, Afifa Zia, Raheel Nawaz, Ayman Fayoumi, Naif Radi Aljohani, and Saeed-Ul Hassan. 2021. Webometrics: evolution of social media presence of universities. *Scientometrics* 126 (2021), 951–967.

[65] Sameerah Talafha and Banafsheh Rekabdar. 2019. Arabic poem generation with hierarchical recurrent attentional network. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*. IEEE, 316–323.

[66] Ken Jon M Tarnate, May M Garcia, and Priscilla Sotelo-Bator. 2020. Short poem generation (spg): a performance evaluation of hidden Markov model based on readability index and turing test. *International Journal of Advanced Computer Science and Applications* 11, 2 (2020).

[67] Berty Chrismartin Lumban Tobing and Ruli Manurung. 2015. A chart generation system for topical metrical poetry.. In *ICCC*. Citeseer, 308–314.

[68] Jukka Toivanen, Oskar Gross, and Hannu Toivonen. 2014. ” The Officer Is Taller Than You, Who Race Yourself!”: Using Document Specific Word Associations in Poetry Generation. In *Proceedings of the Fifth International Conference on Computational Creativity*. Jožef Stefan Institute.

[69] Jukka Toivanen, Matti Järvisalo, and Hannu Toivonen. 2013. Harnessing constraint programming for poetry composition. In *The Fourth International Conference on Computational Creativity*. The University of Sydney.

[70] Jukka Toivanen, Hannu Toivonen, Alessandro Valitutti, and Oskar Gross. 2012. Corpus-based generation of content and form in poetry. In *Proceedings of the third international conference on computational creativity*. University College Dublin.

[71] Nattapol Trijakwanich, Peerat Limkonchotiwat, Raheem Sarwar, Wannaphong Phatthiyaphaibun, Ekapol Chuangsuwanich, and Sarana Nutanong. 2021. Robust fragment-based framework for cross-lingual sentence retrieval. Association for Computational Linguistics.

[72] UKEssays. 2018. Essay, poetry, prose, drama and film in literature. *UK Essays* (November 2018). https://www.ukessays.com/essays/education/essay-poetry-prose-drama-and-film-in-literature.php

[73] Tim Van de Cruys. 2020. Automatic poetry generation from prosaic text. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 2471–2480.

[74] Imke Van Heerden and Anil Bas. 2021. AfriKI: Machine-in-the-Loop Afrikaans Poetry Generation. *arXiv preprint arXiv:2103.16190* (2021).

[75] Martin Tsan Wong, Andy Hon Wai Chun, Qing Li, SY Chen, and Anping Xu. 2008. Automatic haiku generation using VSM. In *WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering*, Vol. 7. Citeseer.

[76] Rui Yan. 2016. i, Poet: Automatic Poetry Composition through Recurrent Neural Networks with Iterative Polishing Schema.. In *IJCAI*, Vol. 2238. 2244.

[77] Rui Yan, Han Jiang, Mirella Lapata, Shou-De Lin, Xueqiang Lv, and Xiaoming Li. 2013. I, poet: automatic chinese poetry composition through a generative summarization framework under constrained optimization. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

[78] Galip ”Umit Yolcu. 2020. Binârî: A Poetry Generation System for Ghazals. (2020).

[79] Xingxing Zhang and Mirella Lapata. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 670–680.

[80] Guo Zhipeng, Xiaoyuan Yi, Maosong Sun, Wenhao Li, Cheng Yang, Jiannan Liang, Huimin Chen, Yuhui Zhang, and Ruoyu Li. 2019. Jiuge: A human-machine collaborative Chinese classical poetry generation system. In *Proceedings of the 57th annual meeting of the association for computational linguistics: system demonstrations.* 25–30.

Appendix: Example Poem and Annotation of Key Terms

Example Poem

<div align="center">

Whispers of the Dawn
The sky blushes as the sun ascends,
Spreading its warmth like a lover's hand.
Dewdrops glisten, jewels in the light,
While shadows flee from morning's might.

</div>

Key Terms and Annotations

- **Verse** (مصرع): A single line in a poem is called a *verse*. Each line in the example poem above represents a verse.
  Example: "The sky blushes as the sun ascends" is the first verse of this poem.
- **Stanza** (قطعہ): A stanza is a grouping of verses in a poem, similar to a paragraph in prose. In this example, all four lines form a single stanza.
  Example: The four lines together form one stanza, with each verse contributing to a complete thought about dawn.
- **Couplet** (شعر): A couplet consists of two verses that rhyme and share the same meter. In this example, the poem contains two couplets, each representing two lines with similar meter and rhyme.
  Example: "The sky blushes as the sun ascends, / Spreading its warmth like a lover's hand" is the first couplet.
- **Rhyme** (قافیہ): Rhyme refers to words with similar sounds, often at the end of verses. In this poem, the rhyme occurs in the last words of each pair of lines.
  Example: "ascends" rhymes with "hand," and "light" rhymes with "might."
- **Refrain** (ردیف): A refrain is a repeated word or phrase that appears after the rhyme in each verse. Although this example poem does not have a refrain, in poetry where a refrain is used, it would appear after each rhyme word to create repetition and emphasis.
  Example: If "dawn" were repeated at the end of each verse, it would be a refrain.
- **Meter** (بحر): Meter is the structured pattern of sounds, determined by stressed (long) and unstressed (short) syllables. The example poem follows an *iambic pentameter* meter, where each line contains five pairs of alternating unstressed and stressed syllables (iambs).
  Example: "The sky blushes as the sun ascends" (five iambs: da-DUM da-DUM da-DUM da-DUM da-DUM).
- **Syllable**: A syllable is a unit of speech sound, consisting of a vowel alone or a vowel with accompanying consonants. Each word in a verse can be broken down into syllables.
  Example: The word "blushes" contains two syllables: "blush" and "-es."
- **Meter Foot**: A meter foot is a combination of syllables in a specific sequence that repeats throughout a line of poetry. In this poem, the meter foot is an *iamb*, which is one unstressed syllable followed by one stressed syllable.
  Example: In the line "The sky blushes as the sun ascends," each pair of syllables like "the sky," "blush-es," and "as the" forms an iamb, which is the repeating meter foot.