Generating Homeric Poetry with Deep Neural Networks

Annie K. Lamar
Department of Classics
Stanford University
Stanford, CA, USA
kalamar@stanford.edu

America Chambers
Department of Computer Science
University of Puget Sound
Tacoma, WA, USA
alchambers@pugetsound.edu

***Abstract*—We investigate the generation of metrically accurate Homeric poetry. We assess the quality of the generated lines of poetry using quantitative metrical analysis and expert evaluation. This evaluation reveals that our model captures complex poetic meter implicitly, but underperforms in terms of semantics and context matching. Our research highlights the importance of expert evaluation and shows that future research should focus on poetic semantics.**

***Index Terms*—poetry generation, neural network, poetic meter, ancient languages**

I. Introduction

Homeric poetry is poetry traditionally ascribed to the bard Homer and includes the *Iliad* and the *Odyssey*. Homeric poetry plays a central role in the field of Classics and strongly influences the study and understanding of Classical literature, mythology, and ancient culture, values, and militarism. In this paper, we investigate the use of deep neural networks for generating metrically accurate Homeric poetry – providing a new pathway for Classicists to analyze the metrical and semantic considerations made in the production of such epic oral poetry.

Generating metrically accurate Homeric poetry is a creative and challenging task. Homeric poetry is written in a strict and complex meter known as *dactylic hexameter*. Most poetry generation models focus either on syllabic-based meters or simple iambic meters. In contrast to such poetic meters, lines of dactylic hexameter contain varying numbers of syllables and words. An additional challenge is the relatively small amount of Homeric poetry available. The entirety of the Homeric canon is about 27,000 lines.

In the next section, we provide a more detailed description of dactylic hexameter. We also discuss the long history of analysis that Homeric poetry has enjoyed. We then provide a description of our data sets and the deep neural network architecture that we use. Finally, we present the results of having expert Classicists evaluate the poetry generated by our system.

II. Homeric Poetry

A. Homer and Oral Poetry

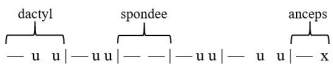
The *Odyssey* and the *Iliad* are two epic poems traditionally ascribed to the bard Homer and dated to the eighth century BCE. The *Iliad* tells the story of the Trojan War and the *Odyssey* recounts the hero Odysseus' long homecoming following the war. Homeric poetry is written in a dialect of ancient Greek called Homeric Greek.

Although we refer to Homer as the author of these texts for convenience, the two works are most likely the product of an extended oral tradition rather than a single poet named Homer. That is, the two works are likely the result of many (unnamed) poets who memorized, performed, and innovated on a set of standard poems over a long period of time.

B. Homeric Meter

Homeric poetry is written in dactylic hexameter, a metrical system in which each line consists of six metrical feet. A single foot can consist of one of three options: a dactyl, a spondee, or an anceps. A dactyl is a foot composed of one long and two short syllables. A spondee is composed of two long syllables. An anceps is composed of one long syllable and one other syllable of either type (long or short). Each line of dactylic hexameter must end with an anceps.

Figure 1 shows the first line of Homer's *Iliad* with the dactylic hexameter marked above. The feet are separated by a vertical bar. A long dash indicates a long syllable, while 'u' indicates a short syllable. In addition, the long syllables in the actual text are bolded. The pronunciation and translation are also shown in English underneath.



μῆνιν ἄειδε θεὰ Πηληϊάδεω Ἀχιλῆος

Mê-nin a-ei-de the-á Pê-lê-i-a-deô A-chi-lê-os

Sing, goddess, the wrath of Achilles, son of Peleus

Figure 1: First line of the *Iliad* with metrical markings.

III. Related Work

A. Related Work in Antiquity

Poets have been attempting to imitate Homeric poetry in its style, content, and meter since ancient times. Plato

978-1-7281-4127-5/19/$31.00 ©2019 IEEE
DOI 10.1109/TransAI46475.2019.00020

writes in the *Republic* that "praisers of Homer... say that this poet educated Greece, and that in the management and education of human affairs it is worthwhile to take him up for study and for living, by arranging one's whole life according to this poet" [1], [2]. Tragic poets contemporary to Plato were indeed considerably influenced by Homeric epic. Homeric motifs and style appear in the tragedies and satyric dramas of Aeschylus, Euripides and Sophocles [3].

Roman poets, including Livius Andronicus, Naevius, Ennius, and Virgil, likewise imitated Homeric epic. The structure of Virgil's *Aeneid* demonstrates this point: the first half of the Virgilian epic reflects the structure of Homer's *Odyssey* while the second half reflects that of the *Iliad* [4]. Homeric poetry influences even (relatively) modern writers; although providing an exhaustive list is beyond the scope of this paper, a few representative examples are James Joyce's *Ulysses*, Margaret Atwood's *The Penelopiad*, William Faulkner's *As I Lay Dying*, and Audrey Niffenegger's *The Time Traveler's Wife*.

In addition, scholars have been attempting to understand and quantify Homeric poetry for millenia. Greek grammatical scholarship from as early as the sixth century BCE has been focused on the reproduction of grammar and syntax of Homeric texts; The sixth century BCE scholar Theagenes of Rhegium, heralded as the first grammarian, is noted for his defense of Homeric grammar and mythology [5]. A couple centuries later, Plato devotes much of his *Republic* to understanding the nature of poetry and uses Homer as a model. In the modern era, we can look to Nietzsche and Freud for philosophical applications of Homer, and Butler and Coleridge for humanistic interpretations of the texts. Homeric texts were read and preserved by a wide range of Mediterranean peoples, from the coast of modern-day Spain, to northern Africa, and into the Near East. Because the transmission of Homeric texts has been considered critical by such a range of cultures and eras, clearly the text communicates universal ideas about the nature of humanity. For this reason, it is worthwhile to apply the tools of modernity to continue attempts to understand and reproduce Homeric poetry.

B. Poetry Generation in Modern Times

In the early twenty-first century most poetry generation was rule-based or statistical in nature.

Oliveira et al. created PoeTryMe, a platform for automatic poetry generation that used grammar rules and templates [7]. Tosa et al. created a rule-based haiku generation system that used arbitrary phrases chosen by a user to produce a haiku with the correct number of syllables on each line [8]. Tosa et al. improved on this rule-based system by incorporating more Japanese cultural characteristics into the generated haikus [9].

Jiang et al. used statistical methods and the principles of machine translation to generate Chinese couplets [14]. He et al. also generated Chinese Classical poems using statistical machine translation models where to generate each output sentence, a model specifically trained for an input sentence is used for generation [15].

Other approaches to poetry generation include genetic algorithms [10], [11], [12] and the application of text summarization techniques [13].

Since 2014, research on automatic poetry generation has been dominated by neural methods. Zhang et al. produced joint character-level recurrent neural networks to better capture poetic content and form [17]. Wang et al. used a bidirectional long short-term memory (LSTM) model to generate Chinese poetry [18]. Ghazvininejad et al. used a dictionary to impose rhyme constraints on poetry generated with a recurrent neural network [19]. In contrast, Lau et al. learned rhyme patterns automatically and proposed a pipeline of language, rhyme, and meter models to generate Shakespearean poetry [20]. Of particular relevance to this paper is Lau et al.'s findings that a vanilla LSTM is able to capture English iambic meter implicitly. In contrast to [20], we attempt to generate dactylic hexameter, a poetic meter more complex than iambic meters.

IV. Datasets

The entirety of the available Homeric canon contains 27,342 lines from Homer's *Odyssey* and Homer's *Iliad*. The text from both works is taken from Oxford's 1920 edition of the *Homeri Operi* [21].

Our training set consists of input output pairs where the input is one line of the poem and the output is the next line of the poem (see Figure 2). For example, given the first line of the *Iliad* as input, the model should generate the second line of the *Iliad* as output, so on and so forth.

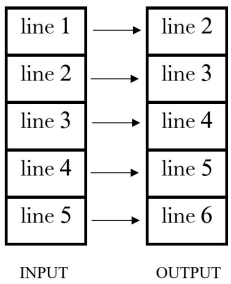


Figure 2: The training set

We generate 30 test sets by following the procedure outlined below 30 times:

- Randomly select a passage of 5 lines from the training set.
- For each passage:
  - Randomly select one line from the passage.
  - Remove the selected line (and its corresponding output) from the training set and place it in the test set.
  - Remove the preceding line (and its corresponding output) from the training set.

– All remaining lines remain in the training set.

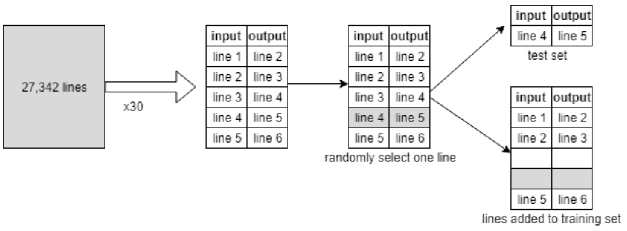


Figure 3: Procedure to generate test sets

This procedure is shown in Figure 3. Note that while the selected line (the shaded row in Figure 3) is added to the test set, the preceding line is not included in either the test set or the training set. The preceding line is removed from the training set so that the selected line (i.e. line 4) does not appear in the training set in any form: neither as an input nor as an output.

Once we have constructed our training and test sets, we further sub-divide the training set to create a validation set. The validation set was constructed by taking the last 10 lines from every 100 lines, resulting in a validation set that was approximately 10% the size of the training set and the reflected the diversity found in the training set.

V. Model Architecture

In this section, we provide a brief overview of the architecture of the recursive neural network (RNN) used in this paper.

A. Encoder-Decoder RNNs

An encoder-decoder RNN [22] is a neural network architecture that maps sequences to sequences. In this case, given a sequence of words (i.e. one line in a poem) the model generates an output sequence of words (i.e. the next line in the poem).

An encoder-decoder RNN works in two stages. The first stage involves mapping the input sequence to a high-dimensional space. This high-dimensional space represents a mathematical encoding of the *meaning* of the sentence that is language independent. The second stage then maps from points in this high-dimensional space back to the space of sequences.

Let $x = [x_1\ x_2 ... x_n]$ be the words in a given line of poetry. The encoder RNN incrementally builds an encoding $\mathbf{h}$ of the input sentence word-by-word using the following algorithm:

// Build the partial encodings
for $t = 1$ to n **do**
  $\mathbf{h}_t = f(\mathbf{h}_{t-1}, x_t)$
end for

//Build the overall encoding
$\mathbf{h} = g(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n)$

where $\mathbf{h}_0$ is typically a vector of all zeros. The partial encodings $\mathbf{h}_t$ capture the meaning of the sentence up to the given word. The final encoding $\mathbf{h}$ is then a function of all of the partial encodings with $g(\cdot)$ commonly chosen to be $g(\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n) = \mathbf{h}_n$. That is, $\mathbf{h}_n$ is assumed to be a sufficient encoding of the entire sentence.

One common choice for the function $f(\cdot)$ is a long short-term memory (LSTM) model [23] that keeps track of an internal state as a form of short-term memory. At each step, the new state is derived from the previous one in a process of forgetting and then updating in response to the new word $\mathbf{x}_t$.

Given the final encoding of the input sentence (which we will now denote as $\mathbf{h}_x$), the next line of the poem $\mathbf{y} = [y_1\ y_2\ ...]$ is generated using another RNN (the decoder). At each step, a word y_t is sampled conditioned on the previously generated word y_{t-1}, the partial encoding $\mathbf{h}_{t-1}$ of the *decoder*, and the encoding of the input sentence $\mathbf{h}_x$. Note that in this context, the partial encoding $\mathbf{h}_{t-1}$ serves as a summary of the meaning of the previously generated words. Algorithmically, we have

while y_{t-1} does not equal `EOS` **do**
  Compute $p(y_t|y_{t-1}, \mathbf{h}_{t-1}, \mathbf{h}_x)$
  Sample $y_t \sim p(y_t|y_{t-1}, \mathbf{h}_{t-1}, \mathbf{h}_x)$
end while

where `EOS` is a specially designated end-of-sentence symbol. The distribution $p(y_t|y_{t-1}, \mathbf{h}_{t-1}, \mathbf{h}_x)$ is, again, commonly chosen to be a LSTM model whose output is passed through the softmax function to produce a valid probability distribution.

In our experiments, we augment the model by using a bidirectional encoding (where we compute partial encodings from the beginning of the sentence forward and from the end of the sentence backward) and an attention mechanism (where the generation of each word y_t is now conditioned on a weighted sum of *all* of the partial encodings of x rather than conditioning on $\mathbf{h}_n$ alone). A bidirectional encoding and attention mechanism have both been shown to significantly improve performance for various natural-language applications [24], [28], [29].

B. Word Embeddings

Our vocabulary is the set of all words found in the *Odyssey* and the *Iliad*. This set of Homeric vocabulary represents a small subset of all ancient Greek diction. Rather than encode each word in the vocabulary as a one-hot vector (i.e., a vector of all zeros whose length is the size of the vocabulary with a single non-zero entry indicating the given word), we follow convention and train GloVe word embeddings [25] using word co-occurences from the entirety of the *Iliad* and the *Odyssey*.

A GloVe word embedding is a mapping from words in the vocabulary to a high-dimensional vector space that is learned from word co-occurrences. The learned word

vectors provide a much richer representation than binary one-hot vectors – e.g., the Euclidean distance between two word vectors provides a measure of the semantic similarity of the words.

VI. Experiments

We use the OpenNMT-py library [26]. The encoder RNN is a 2-layer bidirectional LSTM with encoding dimension of 200 (i.e. the forward and backward encoding each have dimension 100). The decoder RNN is also a 2-layer bidirectional LSTM but with dimension 500. We use pretrained GloVe word embeddings for both the encoder and the decoder. We use an Adam optimizer with dropout probability of 0.3 and an initial learning rate of 0.01. The model was trained for 64,000 iterations at which point the accuracy on the validation set stopped improving.

Once trained, we then applied the model to each of the 30 test sets. For each test set, the input to the model is the selected line of poetry (e.g. line 4 as shown in Figure 3) from which the model generates a new line of poetry. We then assess the generated lines of poetry in two ways: (1) a quantitative metrical evaluation, and (2) an overall evaluation by Classicists to determine how well the generated lines fit into their original passages.

VII. Results

A. Metrical Evaluation

The 30 generated lines of poetry ranged from a minimum of 5 feet to a maximum of 8 feet. All of the generated lines ended in an anceps. Of the total *feet* in the 30 generated lines, 91.3% were correct dactyls, spondees, or anceps. Of the total generated *lines*, 43.33% were in perfect dactylic hexameter.

An example of a generated line with perfect meter is:

ῥηιδίως δαναῶν ἐπιβαινέμεν ἀλλὰ καὶ αυτω

which is composed of a dactyl, dactyl, dactyl, dactyl, dactyl, anceps. Notably, this line demonstrates a phenomenon known as *epic correption* between the last two feet. Epic correption is when a syllable that is typically a long syllable becomes short because the next word begins with a vowel. In this example, epic correption occurs between καὶ and αυτω. This line can be translated as "But the son of Danaoi easily having climbed up, even he..." This is a typical syntax for ancient Greek poetry, which often uses the structure of "But X having happened, Y..." where the action is finished on the next line.

An example of a generated line with poor meter is

ὀφθαλμοῖσι τε κατείρυσται καὶ ἐπαρτέες εἰσὶν ἑταῖροι

which begins with a spondee, then contains a dactyl, and after the second foot becomes unscannable. The translation is also poor, but can be roughly rendered as "both the eyes provoked[1] and enemies are equipped."

[1]"Provoked" is conjugated incorrectly and eyes cannot be the subject of the verb.

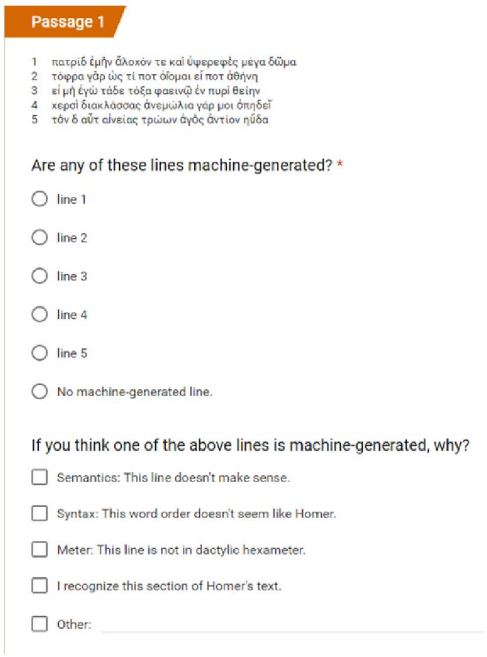


Figure 4: A sample passage from the survey

B. Evaluation by Classicists

To evaluate how well the generated lines fit into the original passages from which they were chosen, we created a survey that displayed the 30 5-line passages of Homeric poetry. Evaluators were instructed that each passage contained *at most* one machine-generated line of poetry (i.e. a passage may not contain any machine-generated lines). The survey contained 27 passages with a machine-generated line and 3 passages with no machine-generated line. For each of the 30 passages, evaluators were asked to identify which line (if any) was machine generated. If they identified a line as machine generated, they were further asked to mark why. There were a total of 10 evaluators all of whom are Classics graduate students. Evaluators were paid $10.00 each for evaluating all 30 passages. An example of a passage from the survey is shown in Figure 4.

We first present the results on a passage level. On the passage level, there are 300 total evaluation instances (10 evaluators and 30 passages). In the contingency table shown in Table I, a true positive occurs when an evaluator tags any line in a passage as machine-generated and that passage contained a machine-generated line. A true positive does not require that an evaluator selected the correct line as machine generated. A true negative occurs when an evaluator correctly states that a passage contains no machine-generated lines. A false positive occurs when an evaluator tags a line in a passage as machine generated, but that passage contains no machine-generated lines. A false negative occurs when an evaluator states a passage contains no machine-generated lines, but the passage does contain a machine-generated line.

	Tagged As MG	Tagged No MG
Contained MG	206	64
Not Contain MG	11	16

Table I: Contingency table for **passage** evaluation

75% of the time evaluators correctly determined if a passage contained a machine-generated line or not [2]. The precision and recall were 94% and 76% respectively with an F_1 score for passage-level evaluation of 0.836. Overall, when an evaluator identified a passage as containing a machine-generated line, they were almost always correct. However, such passages were identified only 76% of the time.

We now present our results at the line level. We consider only results for the 27 passages which contained a machine-generated line. At the line level, there are 1,350 total instances (10 evaluators · 27 passages · 5 lines). Note that these instances are not independent because evaluators were only able to choose a maximum of 1 in every 5 lines. In the contingency table shown in Table II, a true positive occurs when an evaluator tagged a line as machine-generated and the line was machine generated. A true negative occurs when an evaluator indicated a line was not machine-generated (i.e. they did not mark the line in the survey) and the line was not machine generated. A false positive occurs when an evaluator tagged a line as machine generated and it was not. A false negative occurs when evaluators did not mark a line as machine generated, but the line was machine generated.

	Tagged As MG	Tagged No MG
MG Line	171	99
Not MG Line	35	1045

Table II: Contingency table for **line** evaluation

90% of the time evaluators correctly identified a line as machine generated or not. Note that randomly guessing one of the six options (either choosing one of the five lines or "No machine-generated line") would result in an accuracy of 16.67%. The precision and recall were 63% and 83% respectively with an F_1 score for line-level evaluation of 0.71.

Evaluators were also asked to mark the reason why they thought a line was machine generated. They were given five options: (1) Semantics: This line doesn't make sense, (2) Syntax: This word order doesn't seem like Homer, (3) Meter: This line is not in dactylic hexameter, (4) I recognize this section of Homer's text, and (5) Other. Evaluators were allowed to select more than one option. If the evaluator marked "Other," they were asked to type in their own response. Figure 5 below shows the reasons that evaluators marked a line as machine-generated when the line was in fact machine-generated. A majority of the time, evaluators correctly marked a line as machine-generated because semantically the line did not make sense.

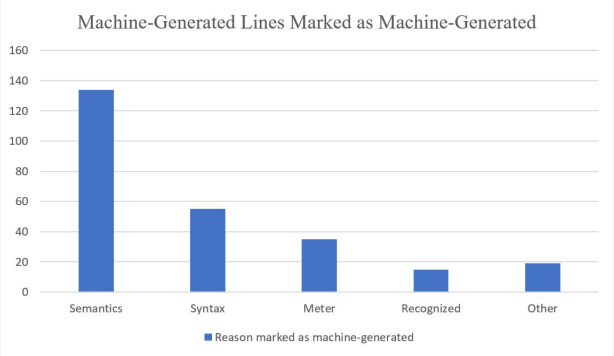


Figure 5: Reasons evaluators selected for why they believed a line was machine-generated.

When evaluators marked their reasoning as "Other," they often presented reasoning related to context – i.e. how the line "fit" with the surrounding lines of poetry. For example, one evaluator writes "[s]emantically, it doesn't seem to necessarily fit with the other lines of text. It may be true Homer, while one of the surrounding lines is machine-generated." Other evaluators write "just guessing but it doesn't seem to fit," "The line doesn't make sense in this context," "Doesn't fit with rest of passage," "Doesn't seem to fit," and "Doesn't fit with rest of text." Another evaluator notes a machine-generated line "seems to interrupt meaning between lines 3 and 5."

Another common thread in "Other" responses is evaluators' insider knowledge of Homeric poetry being used to identify machine-generated lines even when those lines were semantically and metrically sound. For example, one evaluator notes that "[l]ines ending in glaukopis Athene tend to be self-contained units where Athena performs the action of a verb." Here, the student was only able to identify the line as machine-generated because they were aware of how the Greek formula *glaukopis Athene* is used in Homeric poetry. Another evaluator writes "almost never see 'pallas' by itself, normally is 'pallas athene.' In this case, the evaluator was able to tell the line was machine-generated because of their familiarity with the Homeric formula *Pallas Athene*.

Other times, evaluators were able to identify machine-generated lines because of their knowledge about Homeric characters roles within texts. For example, consider the lines below:

αὐτὰρ ἐπεὶ δὴ δούρατ ἀλεύαντο μνηστήρων
ἷξε τόθ ἔσαν οἶκον δὲ καλυψώ δῖα θεάων

But when they had avoided the spears of the suitors,
He came then to the house of Calypso heavenly among

[2]Although they did not necessarily correctly identify which line was machine generated.

goddesses.

The first line is true Homeric poetry from Homer's *Odyssey*. The second line is machine-generated. In the *Odyssey*, the suitors (μνηστήρων) are a group of characters who never come in contact with the character Calypso (καλυψώ). Evaluators noticed this. They wrote reasons such as "Calypso should not be involved in this context with suitors" and "Calypso and the suitors? I'd watch that show, but nope!"

The same type of error occurs in a second passage involving the characters Penelope and Hector. Penelope is a character who remains on the island of Ithaca for the duration of the storyline told by Homer's *Iliad* and *Odyssey*. Hector is a character who remains at Troy during most of the *Iliad* and is killed partway through the epic. These two characters clearly could not have ever occupied the same location at any point in the Homeric canon. However, a line including Hector is generated following a line about Penelope. In the below lines, the first three are Homeric poetry while the last line is machine-generated:

θηρσὶ καὶ οἰωνοῖσιν ἕλωρ γένετ οὐδέ ἑ μήτηρ
οὐδ ἄλοχος πολύδωρος ἐχέφρων πηνελόπεια
ἕκτορα ἅντα κατ ὅσσε παρίστατο δάκρυ χέουσα

Nor did his mother deck him for burial and weep over him,
Nor his father, we who gave him birth, no, nor did his wife,
wooed with many gifts, constant Penelope.
Face-to-face with Hector she stood and both eyes shed tears.

The reasons graduate students provided for identifying the latter line as machine-generated included "Hector should not be involved in this context" and "Get out of here, Hector."

Sometimes evaluators marked true lines of Homer as machine-generated. Figure 6 shows the reasons graduate student provided for marking a line as machine-generated when the line was not machine-generated.

The most common reason that a true line of Homeric poetry was marked as machine-generated was, again, semantics. This reasoning has no correlation to the placement of the machine-generated line in the passage being placed immediately before or after the line selected by the evaluator. Evaluator's confusion was likely because they were reading passages out of context.

Evaluators also marked meter as the reason they believed a real line was machine-generated. The comments left in the "Other" category suggest that evaluators may have marked "Meter" because the Homeric line contained rare (but correct) metrical patterns. For example, an evaluator marked this line of Homeric poetry as machine-generated:

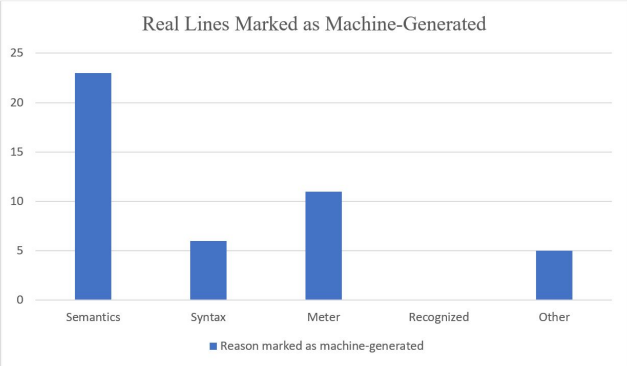


Figure 6: Reasons evaluators selected for why they believed a line was machine-generated.

ὣδε δέ τις εἴπεσκεν ἀχαιῶν τε τρώων τε

The above line is scanned dactyl-spondee-dactyl-spondee-spondee-anceps. This evaluator commented that "it is rare for a spondee to appear in the fifth foot of dactylic hexameter." The presence of a spondee in the penultimate foot of a hexmetric line is unusual, although not unheard of. The presence of a spondee in the antipenultimate foot also makes the penultimate spondee more common and characterizes the line as 'spondeic.'

In general, the evaluators were able to identify machine-generated lines of Homeric poetry primarily for semantic and syntax reasons. If we consider only the lines identified by evaluators as machine-generated based upon metrical reasons, the evaluator accuracy drops to 12.96%. Based on these results, we conclude that we are able to successfully generate poetry in dactylic hexameter.

C. Iliad Generation Experiment

Finally, to investigate the ability of our model to produce entire passages of poetry in dactylic hexameter (as opposed to generating single lines), we generate the first seven lines of the *Iliad*. We re-train our model using a modified training set where the first seven lines of the *Iliad* were removed. Using our model, we use line 1 of the *Iliad* to generate line 2 of the output. We then use the machine-generated line 2 to generate line 3 of the output, and so on.

The lines below are the true first seven lines of Homer's *Iliad*. The translation of the true text of the Iliad is by Caroline Alexander [27].

Homer

μῆνιν ἄειδε θεὰ Πηληϊάδεω Ἀχιλῆος
οὐλομένην ἣ μυρί Ἀχαιοῖς ἄλγε ἔθηκε
πολλὰς δ ἰφθίμους ψυχὰς Ἄϊδι προΐαψεν
ἡρώων αὐτοὺς δὲ ἑλώρια τεῦχε κύνεσσιν
5οἰωνοῖσί τε πᾶσι Διὸς δ ἐτελείετο βουλή
ἐξ οὗ δὴ τὰ πρῶτα διαστήτην ἐρίσαντε

Ἀτρεΐδης τε ἄναξ ἀνδρῶν καὶ δῖος Ἀχιλλεύς

[1] Wrath-sing, goddess, of the ruinous wrath of Peleus' son Achilles
[2] that inflicted woes without number upon the Achaeans
[3] hurled forth to Hades many strong souls of warriors
[4] and rendered their bodies prey for the dogs
[5] for all birds, and the will of Zeus was accomplished
[6] sing from when they too first stood in conflict-
[7] Atreus' son, lord of men, and godlike Achilles.

The lines below are the machine-generated Greek text and corresponding translation. The translation of the machine-generated lines was completed by the author. In the translation of the machine-generated lines, text placed within brackets does not fit grammatically into the sentence. In addition, we use the notation "[→ X]" to indicate that a line does not flow logically with the next machine-generated line of poetry.

Machine-Generated Passage

μῆνιν ἄειδε θεὰ Πηληιάδεω Ἀχιλῆος
αἴθ ὄφελες παρὰ νηυσὶν ἐπεί περ ἐὼν αὐτὸν
τειρομένοις ἑτάροισιν ἀμυνέμεν ἔνθεν ἀπῆλθεν
ἀντίλοχος μεγάλη δὲ ποθὴ πυλίοισιν ἐτύχθη
ἀλλ ὅ γε τοῖσιν μὲν θρασυμήδεα δῖον ἀνῆκεν
αὐτὸς δ αὖτ ἐπὶ πατρόκλῳ ἥρωι βεβήκει
στῆ δὲ παρ αἰάντεσσι θέων εἴθαρ δὲ προσηύδα

[1] Wrath-sing, goddess, of the ruinous wrath of Peleus' son Achilles
[2] If only you paid him alongside the ship then, even though being [→ X]
[3] To defend your hard-set comrades when he goes away, [→ X]
[4] Antilochus, [great longing] was made ready for the Pylians.
[5] And so he, with them, was reaching the heavenly Son of Nestor,
[6] But he had advanced again in honor of the hero Patroclus.
[7] He stood alongside Ajax, but at once he spoke [of the goddesses]

There are a few points of interest in these generated lines. First, Antilochus is first mentioned in line 4 and then mentioned again in line 5 as "the Son of Nestor". Second, Patroclus and Ajax are two of the central heroic figures in the *Iliad*, and so it makes sense that they both would appear in the same context in lines 6-7.

In addition, line 2 begins "If only you had paid him alongside the ship." Much of the machine-generated language associated with the character Achilles is related to what Achilles is owed. This makes sense given that the language in the beginning of the *Iliad* references what debts ought to be paid to Achilles after he sacked the city of Chryse.

In terms of a metrical analysis, the following were used to evaluate the quality and accuracy of the meter of the generated lines:

1) The number of lines (out of 6) that had six total feet.
2) The number of lines (out of 6) that had an anceps.
3) The number of lines (out of 6) that were in perfect dactylic hexameter.
4) The percentage of feet in all 6 lines that were scannable (i.e. a correct dactyl, spondee, or anceps).

The metrical results of our *Iliad* generation experiment are summarized in Table III below. Note that there are a total of 6 machine-generated lines.

Lines with 6 feet (%)	83.33
Lines with ancipites (%)	100
Feet correct (%)	91.89
Lines perfect (%)	66.67

Table III: Metrical evaluation of the machine-generated lines of the *Iliad*

As Table III shows, the majority of feet (92%) are correctly generated and the majority of lines (83%) have the correct number of feet. Overall, two-thirds of machine-generated lines from this *Iliad* generation experiment are perfect dactylic hexameter.

VIII. Conclusion

In this paper, we investigated the use of encoder-decoder RNNs for generating metrically accurate Homeric poetry. Our machine-generated lines of poetry were evaluated both metrically and in their original contexts by Classicists. The survey results revealed that our model underperforms in terms of generating semantically cohesive lines of poetry that fit the broader context. However, the model is quite successful in generating Homeric poetry that is metrically accurate.

IX. Future Work

In the future, we will focus on generating lines that are more semantically cohesive and fit within the larger context of the poem. One straightforward method for doing so is to consider more than just the previous line of poetry when generating the next line – e.g., given lines 1 *and* 2, generate line 3 in the poem.

In addition, Homeric characters often appeared in passages together when they should not. This suggests that we need a model that allows us to control where characters appear within the poem. Since an encoder-decoder RNN produces a probability distribution for the next word in the poem, we plan to investigate methods for biasing these distributions towards words that are likely given both the location of the line within the poem as well as the location of the word in the line.

X. Acknowledgments

The authors of this paper would like to thank the following people for their help evaluating machine-generated Homeric poetry: Emory B. Brigden from the University of Puget Sound; Konnor Clark, Edgar A. Garcia, and Megan O'Donald from the University of Washington; Rachel E. Dubit, Grace Erny, Nick Gardner, Dillon Gisch, and Brian D. Le from Stanford University; Kathryn H. Stutz from John Hopkins University.

References

[1] Plato. *The Republic*. Robin Waterfield, Trans. Oxford and New York: Oxford University Press, 1993, sec. 606e1-5.

[2] C.L. Griswold. *Plato on Rhetoric and Poetry*, in *The Stanford Encyclopedia of Philosophy*, E.N. Zalta, ed. 2016. [Online]. Available: https://plato.stanford.edu/archives/fall2016/entries/plato-rhetoric. [Accessed April. 24, 2019].

[3] J.E. Sandys. *History of Classical Scholarship From the Sixth Century B.C. to the End of the Middle Ages*. Cambridge: Cambridge University Press, 1903.

[4] Maurus Servius Honoratus. *Vergilii carmina comentarii*, in *Servii Grammatici qui feruntur in Vergilii carmina commentarii*, G. Thilo and H. Hagen, Eds. Leipzig: B.G. Teubner, 1881.

[5] J.I. Porter. *Homer, Skepticism, and the History of Philology*, in *Modernity's Classics*, S.C. Humphreys and R.G. Wagner, Eds. Oxford and Heidelberg: Springer, 2013, pp. 261-292.

[6] H.G. Oliveira. "Automatic Generation of Poetry: An Overview," 2009.

[7] H.G. Oliveira. "PoeTryMe: A Versatile Platform for Poetry Generation," In *Proc. ECAI 2012 Workshop on Computational Creativity, Concept Invention, and General Intelligence*, 2012, pp. 1-21.

[8] N. Tosa, H. Obara, and M. Minoh. "Hitch Haiku: An Interactive Supporting System for Composing Haiku Poem," In *Entertainment Computing*, ICEC, 2008, pp 209-216.

[9] N. Tosa, X. Wu, and R. Nakatsu. "New Hitch Haiku: An Interactive Renku Poem Composition Supporting Tool Applied for Sightseeing Navigation System," in *Entertainment Computing*, ICEC, 2009, pp. 191-196.

[10] H.M. Manurung. "An Evolutionary Algorithm Approach to Poetry Generation," *Dissertation, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh*, 2003.

[11] C. Zhou, W. You, and X. Ding. "Genetic Algorithm and Its Implementation of Automatic Generation of Chinese SONGCI: Genetic Algorithm and Its Implementation of Automatic Generation of Chinese SONGCI," in *Journal of Software* 21.3, pp. 427-437, 2010.

[12] R. Manurung, G. Ritchie, and H. Thompson. "Using Genetic Algorithms to Create Meaningful Poetic Text," in *Journal of Experimental & Theoretical Artificial Intelligence*, pp. 43-64, 2010.

[13] R. Yan, H. Jiang, M. Lapata, S. Lin, X. Lv, and X. Li. "i, Poet: Automatic Chinese Poetry Composition through a Generative Summarization Framework under Constrained Optimizationi" in *Proc. of the Twenty-Third International Joint Conference on Artificial Intelligence*, 2013, pp. 2197-2203.

[14] L. Jiang and M. Zhou. "Generating Chinese Couplets Using a Statistical MT Approach," in *Proc. 22nd International Conference on Computational Linguistics*, 2008, pp. 377-384.

[15] J. He, M. Shou, and L. Jiang. "Generating Chinese Classical Poems with Statistical Machine Translation Models," in *Proc. Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2014, pp. 1650-1656.

[16] E. Greene, T. Bodrumlu, and K. Knight. "Automatic Analysis of Rhythmic Poetry with Applications to Generation and Translation," in *Proc. 2010 Conference on Empirical Methods in Natural Language Processing*, ACL, 2018, pp. 524-533.

[17] X. Zhang and M. Lapata. "Chinese Poetry Generation with Recurrent Neural Networks," in *Proc. 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP, 2014, pp. 670-680.

[18] Q. Wang, T. Luo, D. Wang, and C. Xing. "Chinese Song Iambics Generation with Neural Attention-Based Model," in *Intl. Joint Conf. on Artificial Intelligence*, 2016, pp. 2943-2949.

[19] M. Ghazvininejad, X. Shi, Y. Choi, and K. Knight. "Generating topical poetry," in *Proc. 2016 Conference on Empirical Methods in Natural Language Professing*, Association for Computational Linguistics, 2016, pp 1183-1191.

[20] 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 *Proc. 56th Annual Meeting of the Association for Computational Linguistics*, ACL, 2018, pp. 1948–1958.

[21] Homer. *Homeri Opera in Five Volumes*. Oxford, Oxford University Press: 1920.

[22] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in *Proc. of the 2014 Conf. on Empirical Methods in Natural Lang. Processing (EMNLP)*, 2014, pp. 1724-1734.

[23] S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory," in *Neural Computation* 9.8, pp. 1735-1780, 1997.

[24] M. Schuster and K.K. Paliwal. "Bidirectional Recurrent Neural Networks," in *IEEE Transactions on Signal Processing* 45.11, 1997, pp. 2673-2681.

[25] J. Pennington, R. Socher, and C.D. Manning. "GloVe: Global Vectors for Word Representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532-1543.

[26] G. Klein, Y. Kin, Y. Deng, J. Senellart, and A.M. Rush. "OpenNMT: Open-Source Toolkit for Neural Machine Translation." *Arxiv*: 1701.02810, 2017.

[27] C. Alexander, Trans. *The Iliad: A New Translation*. New York, Harper Collins: 2015.

[28] D. Bahdanau, K. H. Chou, and Y. Bengio. "Neural machine translation by jointly learning to align and translate." *Arxiv*: 1409.0473, 2016.

[29] M. T. Luong, H. Pham, and C. Manning. "Effective approaches to attention-based neural machine translation." *Arxiv*: 1508.04025, 2015.