CS230

Image to Ballad Generator

(Generative Modeling, Natural Language Processing)

Maciej Kurzynski*
East Asian Languages and Cultures
Stanford University
makurz@stanford.edu

Mai Lan Nguyen*
Department of Computer Science
Stanford University
nmailan@stanford.edu

Zuyi Liz Zhao*
Department of Computer Science
Stanford University
zuyi@stanford.edu

1 Introduction

Generating metrically accurate ballads is a creative and challenging task. A typical ballad consists of stanzas that form quatrains, or four poetic lines. Usually, only the second and fourth lines are rhymed (in the scheme ABCB or ABAB). The high complexity of creative language as well as genre and rhyming constraints create substantial challenges for poetry generation for both humans and machines.

While there are numerous projects on generating free-form poetry given visual/image input or a stylistic blueprint, little work has been done on generating poems that specifically fulfill the constraints of a ballad given a particular image. In this paper, we investigate the potential of using **three CNNs and GPT-2** for generating **metrically accurate ballads** that are thematically connected with a visual **(image) input**.

2 Related work

Attempts towards poetry generation have mostly focused on generating poetry that follows a specific author's poetic style [1, 2], syllabic and metric rule, and rhyme scheme [3]. More recently, work on poetry generation has included multi-modal work in producing poetry from image input, such as the work by [4]. In these attempts, the utilization of RNNs or LSTMs proved to be successful in producing poetry that fulfilled their specific genre constraints. However, these methods still have the problem of topic drift and semantic inconsistency with the input, and the image-poem pairs dataset is hard to be built when training these models.

On the other hand, Liu et al.[5] were able to produce fairly semantically-consistent poems, by casting the problem in a multi-adversarial procedure and use CNN-RNN generative model as an agent. Nevertheless, they were only free-verse and were not constrained to fulfilling concrete genre (meter or rhyme-scheme) specifications.

Although much has been done in the area of poetry generation, to our knowledge, poetry generation from image input has either failed on the semantic or the genre requirement. In particular, generating metrically-accurate ballads that are semantically-consistent with some input image has been relatively unexplored. As such, we look toward work in generating ballads that satisfy both relevance to the image and poeticness in language level.

*Denotes equal contribution. Author ordering is alphabetical.

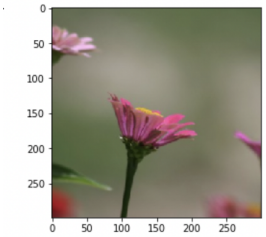
CS230: Deep Learning, Fall 2022, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

To train our model, we used two datasets: the MultiM-Poem dataset developed by Liu et. al[5] and a dataset of 6597 ballads from the UCSB English Broadside Ballad Archive [6].

3.1 MultiM-Poem Dataset Preprocessing

The MultiM-Poem dataset from Liu et. al.[5] is a collection of 8,292 poem-image pairs, with an average of 7.2 lines per poem and 5.7 words per line. The poems in the MultiM-Poem dataset are free-verse poems: they do not necessarily rhyme or follow syllable constraints.



what is lovely never dies
but passes into other loveliness
star-dust or sea-foam flower or winged air

To prepare the dataset for training, we first removed invalid image URLs, leaving us with 7704 images. We then normalized the images by downloading them from their URLs, resizing and center-cropping them to 299x299 pixels, and dividing the image array values by 255. This standardization process ensured that all of the images were the same size and had similar pixel values, which improved the performance of the model.

Each image was associated with at least one free-verse poem. We pre-processed each poem using spaCy POS tagger [7] and NLTK [8] WordNet [9] tool to extract abstract nouns which we used as image scene labels, physical nouns - used as object labels, and the adjectives - as sentiment labels. The labels associated with each image were converted to multi-hot vectors. The labels were imbalanced, but due to time constraints, we did not perform data augmentation to rectify this. After pre-processing and normalization, we split the images into training, validation, and test sets with a 80-10-10 ratio, respectively. This was done three times, once for each label type.

3.2 Ballad Dataset

To create training examples for fine-tuning our GPT-2 model, we split the ballads into quatrains that followed one of the four rhyme schemes (AABB, ABCB, ABAC, ABAB). We then extracted keywords related to objects, scenes, and sentiments depicted in each full ballad. We used the spaCy POS tagger and NLTK WordNet to extract nouns and adjectives, given that objects and scenes are usually conveyed through nouns, whereas sentiments – through adjectives. Next, we used WordNet to find synonyms for these keywords and randomly selected three synonyms for each label (object, scene, sentiment) for each quatrain. By identifying synonyms, we prevented the model from reusing the keywords from the prompt in the generated ballad. Finally, we extracted four rhymes from each quatrain and appended them to the training prompt. This gave each quatrain a total of 13 (3 * 3 + 4) closely associated label words.

We used the NLTK edit distance metric to approximate corrections for the Old English spellings in our ballads that would not otherwise be recognized by the spaCy POS tagger and NLTK WordNet. The edit distance measures the similarity between two strings by counting the minimum number of single-character edits needed to transform one string into the other. We used the NLTK Brown corpus as a list of valid words to compare against unrecognized words.

4 Methods

4.1 Baseline

The architecture of our baseline is as follows:

1. InceptionV3 classifies the given image. The classification is stored as a **keyword**.
2. Using the ConceptNet API, we find the top 3 **related terms** to the keyword.
3. The 3 related terms are then given to the GPT-2 generator as part of **a prompt**.

Taking inspiration from Loller-Anderson and Gambäck's design, [4] we used ConceptNet[10] and NLTK's CMUDict[8] to identify related terms and count syllables, respectively. We also used the pronouncing[11] module in conjunction with custom code accessing NLTK's CMUDict in order to accurately count syllables.

For the poetry generation, we used the publicly available pre-trained OpenAI GPT-2 English model. We finetuned the model on our ballads datasets by feeding it a prompt consisting of three semantic keywords, which were selected randomly from the quatrain to be reconstructed.

4.2 Model

Our image-to-poem generator model combines three convolutional neural networks (CNNs) and a GPT-2 model to generate ballad quatrains from input images.

Inspired by Liu et. al's design,[5] we chose to utilize 3 CNNs, each fine-tuned to recognize specific artistic elements of an image, in order to increase the content similarity between an input image and the generated poem from our baseline, which only used one pre-trained CNN.

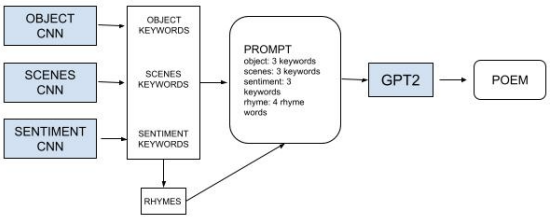


Figure 1: 3CNN-GPT-2 Model

The 3 CNNs are based on the InceptionV3 architecture, which is pre-trained for single-class image classification. Using transfer learning, we adapted the CNNs for the task of multi-label classification. We added additional 4 layers (GlobalAveragePooling2D, Dense, Dropout, and an output Dense) and changed the output activation from softmax $\texttt{softmax}(x) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$ to sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$.

From the preprocessed MultiM-Poem dataset, we obtained three types of labels: objects, scenes, and sentiments—and associated each type with a different CNN. We fine-tuned each model to learn to label images according to its corresponding label type. We fine-tuned each CNN model using binary cross-entropy as the loss function and precision as the evaluation metric. The binary cross-entropy loss equation for multi-labeling is $J(w) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{C} y_{n,i}\log(\hat{y}_{n,i}) + (1-y_{n,i})\log(1-\hat{y}_{n,i})$. Precision is defined as the the number of true positive predictions made by the classifier, divided by the total number of positive predictions made by the classifier. We set the top_k value of the precision metric to the number of labels of the corresponding type.

Each CNN was trained for 5 epochs using the Adam optimizer, with a learning rate of 0.001 and beta values of 0.9 and 0.999. Due to memory constraints, we used a batch size of 1. Due to the large number of labels per type and the imbalance in the dataset, our CNN models had low precision scores. To improve their performance, we processed the output of each model by selecting the top 10 highest-value labels and randomly selecting 3 labels. This allowed us to retrieve 9 keywords: 3 per each of the 3 CNNs for each image input.

In fine-tuning, we use the CNN-generated keywords. We find their synonyms with NLTK and sample them at random to generate rhymes following one of the accepted rhyme schemas (ABAB, AABB, ABCB, ABAC). We combine the CNN keywords and the rhymes to form a prompt:

```
<|beginoftext|>objects: [3 keywords]
scenes: [3 keywords]
sentiments: [3 keywords]
rhymes: [4 words]
ballad: [ballad]
<|endoftext|>
```

The used GPT-2 model was fine-tuned on our ballads dataset using the sparse categorical entropy loss: $J(w) = -\frac{1}{N}\sum_{i=1}^{N}[y_i\log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$. We used the Adam optimizer with

exponential decay learning schedule with decay rate of 0.8. The batch size is 32 ballads due to memory constraints. We start fine-tuning with the learning rate of 0.0001. Our GPT-2 English model follows the OpenAI architecture: 24 decoder layers, embedding size of 1024 and 16 self-attention heads, with a language modeling head on top (linear layer with weights tied to the input embeddings).

4.3 Experiments

While we worked on our CNN multi-label classifiers, we experimented with different performance metrics, including Hamming loss and precision. We found that using precision improved the model's performance. Due to our large number of labels for each CNN, our models had low performance. We experimented with different label filtering cutoffs, such as getting the top n labels, but this reduced the number of usable images and diminished semantic flexibility.

To improve on our baseline in generating rhymes, we initially experimented with redefining the GPT-2 loss function in the language modeling head to be a sum of the original GPT-2 Sparse Categorical Cross Entropy Loss and the rhyming loss, which we defined as the Sigmoid Cross Entropy With Logits. We chose this as the rhyming loss because to find a rhyming word is a multi-label classification problem – for any word, there is n number of words (where n is between 1 and the size of vocabulary) that rhyme with it. However, this number is usually much smaller then the total size of the vocabulary (i.e., positive classes are underrepresented). As a consequence, the model can learn to predict 0 for each label and significantly reduce loss, without ever producing correct results. We thus experimented with negative sampling (for n rhyming words, we chose n non-rhyming words as negative examples for each rhyme pair). The logistic loss is:

$$J(w) = L_{SCL} + L_{rhyme} \tag{1}$$

$$= -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y_i}) + (1-y_i)\log(1-\hat{y_i})\right] - \left(z\log(y) + (1-z)\log(1-y)\right) \tag{2}$$

Where $\hat{y}$ are the logits, y are the language labels, and z are the rhyming labels.

This approach, however, failed, as we found that combining the rhyming loss with the language modeling loss is a difficult multitask learning problem given their different scales. The sparse categorical cross entropy loss is computed for all non-masked words in the batch, whereas sigmoid cross entropy loss is computed only for the ending words in each quatrain. We attempted to solve this problem by hyperparameter tuning, multiplying the rhyming loss by a constant (e.g., by the total number of non-masked words in the batch). Nevertheless, while the overall loss was decreasing in this approach, the model still did not learn to produce satisfying rhymes.

We hypothesize that this is due to divergent objectives of the two summed losses – the causal language modeling loss focuses on semantics, whereas the rhyming loss focuses on phonetics, i.e., two words might rhyme despite never appearing in the same semantic context. This ultimately leads to discrepancy between syntactical structure and semantic coherence of the generated poems. This issue was manifested in a generation of poems that ended each line with the word "the" to optimize simultaneously for rhyming loss and language modeling loss ("the" being a common word).

Ultimately, our experimentation with redefining the loss function for GPT-2 was not as effective as our prompt-based method, proposed in section **4.2**.

5 Results and Discussion

We evaluated the performance of our naive baseline model and final 3CNN-GPT-2 model by assessing the quality of the generated ballads through three key metrics: 1) the ballads' line-syllable count 2) their conformity to a rhyme scheme (one of ABAB, ABCB, ABAC, AABB), and finally 3) their average ConceptNet relatedness score. We found that our 3CNN and GPT-2 model improves upon the baseline significantly in generating rhyming poems as well as retaining semantic integrity. However, it performs much worse in creating verses that follow a strict syllable count.

Out of all poems generated by the baseline model, around 1% of lines have exactly 14 syllables and around 20% contain 10-14 syllables. In contrast, our 3CNN-GPT-2 model produces ballads where nearly no lines have 14-syllables and only 0.5% of lines have 10-14 syllables:

Model	% lines with 14 syllables	% lines with 10-14 syllables
baseline2	1.9%	26%
3CNN-GPT-2-med	0.03%	0.8%

In contrast to our baseline model, which completely underperform in following any rhyme schemes or producing any rhymes, our 3CNN-GPT-2 model manages to generate ballads, out of which around 65.3% have any rhymes. We have identified that, it is most successful in producing poems that follow the AABB rhyme scheme (7% of generated poems).

Model	% ABAB	% ABCB	% ABAC	% AABB	% any rhyme
baseline2	0.0%	0.9%	0.9%	0.0%	6.1%
3CNN-GPT-2-med	10%	23.5%	21.4%	4.1%	65.3%

Moreover, our 3CNN-GPT-2 model also outperforms the baseline in generating poems that are semantically consistent with keywords generated from provided images by an entire degree of magnitude. The ConceptNet scores, which measure relatedness (with 1 being most related and 0 least related) are summarized in the table below:

Model	ConceptNet Score
baseline2	0.024
3CNN-GPT-2-med	0.228

Through our qualitative analysis, we found that our model produces poems that have better grammatical and logical integrity than baseline2 generates poems. And while the keywords generated from images don't consistently retain semantic integrity with the images, they are better in our model than the baseline. (See extra appendix below for examples of generated poems, keywords and their corresponding images).[2]

The increase in the number of rhyming lines in the generated quatrains was mainly due to the presence of rhyme schemas in the prompts. It seems that without specifically indicating the rhyming words, the model remains unable to learn rhyming, as the rhyming signal is too weak in the medium of text. The failure of us getting the right syllable count, in turn, was most likely due to the used dataset, whose ballads rarely hit that syllable count. As hypothesized, the added complexity of using three CNNs instead of one for extracting different semantic aspects of the input images significantly improved the semantic extraction of the images and helped us produce better related poems.

6 Conclusion and Future Work

The existing approaches for image to ballad generation utilize regular simple LSTM-RNNs for text generation [4], whereas GPT2 has not been used. The novelty of our project then is in using existing approaches for extracting semantic content of the image with using GPT2 for text generation, as well as experimenting with GPT2 architecture itself to achieve better ballad form-conformity (e.g. redefining loss function to achieve better rhyme/syllable performance or designing a prompt that would force the GPT2 to learn rhymes).

We found that our prompt-based approach to poetry generation with GPT-2 significantly outperforms a simple single CNN and GPT-2 baseline. However, our attempts to redefine loss to force the model to produce rhymes were unsuccessful due to the conflicting objectives of the rhyming loss and the general GPT-2 loss.

Training language models such as GPT-2 to produce metrically rigorous poetry that is semantically related to visual input is difficult because of the lack of consistency in the poems in available datasets. Poems rarely follow its genres specifications strictly and so we can't consistently achieve correct syllable counts or rhyme schemes.

For future work, we hope to create better datasets with ballad-image pairs and standardized ballads (in terms of spelling, syllables, and rhyme scheme). We would better address issues of dataset imbalance. In addition, we hope to reinvestigate ways to implement syllable restrictions in poem generation.

[2]Our experiments included an additional baseline (which was trained on a smaller dataset of 117-metrically rigorous ballads) and a smaller GPT-2 model, whose results you can see in the appendix.

7 Contributions

Mai Lan worked on GPT2 experimentation, combining the two models for poetry generation, developing the evaluation metrics, developing code for JSON file parsing, syllable analysis, rhyme analysis and ConceptNet score, as well as providing model performance analysis.

Zuyi finetuned the 3 CNNs for multi-label classification and created the training prompt generator for post-processing the output of the CNNs. Zuyi also worked on the code interacting with ConceptNet, the NLTK corpora (WordNet, CMUDict, Brown), and the spaCy POS tagger. Zuyi further contributed to ballad dataset preprocessing and developed code for JSON file parsing and syllable analysis.

Maciej worked on collecting and preprocessing the ballad dataset, multitask learning experiments, finetuning the GPT model, and hyperparameter search.

All team members worked on the conceptual development of the methods.

References

[1] Annie Lamar and America Chambers. Generating homeric poetry with deep neural networks. In *2019 First International Conference on Transdisciplinary AI (TransAI)*, pages 68–75, 2019.

[2] Andrea Zugarini, Stefano Melacci, and Marco Maggini. Neural poetry: Learning to generate poems using syllables. 2019.

[3] Cole Peterson. Generating rhyming poetry using lstm recurrent neural networks. 2016.

[4] Malte Loller-Andersen and Björn Gambäck. Deep learning-based poetry generation given visual input. In *ICCC*, 2018.

[5] Bei Liu, Jianlong Fu, Makoto P. Kato, and Masatoshi Yoshikawa. Beyond narrative description: Generating poetry from images by multi-adversarial training. In *2018 ACM Multimedia*, pages 783–791. ACM, October 2018.

[6] Patricia Fumerton. English broadside ballad archive.

[7] Matthew Honnibal, Ives Montani, Sofie Van Landeghem, and Adriane Boyd. spacy: Industrial-strength natural language processing in python. 2020.

[8] Steven Bird, Edward Loper, and Ewan Klein. *Natural Language Processing with Python*. O'Reilly Media Inc, 2009.

[9] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[10] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. *CoRR*, abs/1612.03975, 2016.

[11] Allison Parrish. pronouncing.

8 Appendix: Full Model Results, Poems and Images

8.1 Full Model Results

baseline1 - trained on a smaller dataset of 117 ballads (taken from PoetryFoundation under the form tag for ballads) that strictly follow the ballad structure

baseline2 trained on the larger dataset that contains ballads that are looser in form

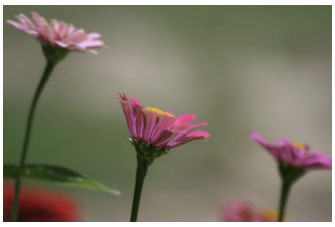
Model	% lines with 14 syllables	% lines with 10-14 syllables
baseline1	0.8%	17%
baseline2	1.9%	26%
3CNN-GPT-2	0.0%	0.6%
3CNN-GPT-2-med	0.03%	0.8%

Model	% ABAB	% ABCB	% ABAC	% AABB	% any rhyme
baseline1	0.0%	0.9%	1.8%	0.0%	6.1%
baseline2	0.0%	0.9%	0.9%	0.0%	6.1%
3CNN-GPT-2	0%	1%	0%	6.1%	40%
3CNN-GPT-2-med	10%	23.5%	21.4%	4.1%	65.3%

Model	ConceptNet Score
baseline1	0.0204
baseline2	0.024
3CNN-GPT-2	0.05
3CNN-GPT-2-med	0.228

8.2 3CNN-GPT2 Poems

8.2.1 Poem 1



Object keywords: "sky", "tree", "rain"
Scene keywords: "love", "day", "life"
Sentiment keywords: "many", "green", "good"
Rhymes: "love", "dove", "pelting", "smelting"

I'll hug thee and kiss thee, and bid you love,
I'll dance with thee thy pretty Turtle Dove.
Thou Youth, as weasse from theee a spinning, a Blot, or Smel
I Shall I stay, shall I not, shall my spirit be seen a Wetting h! phill

8.2.2 Poem 2

Object keywords: "trees", "light", "sky"
Scene keywords: "heart", "morning", "silence"
Sentiment keywords: "many", "little", "green"
Rhymes: "sky", "mai", "little", "whittle"

"Like Phoebus clouded in the Sky,
All glooming were the Shines of Love's Lye:O're spread ore the Earth by those that lie
Heart-straine on the Drapers of that Pitch;
Thy fairest eyes are like the Vlich"

8.2.3 Poem 3

Object keywords: "flower", "leaves", "sun"
Scene keywords: "silence", "life", "dark"
Sentiment keywords: "own", "bright", "green"
Rhymes: "life", "wife", "own", "moan"

For thou shalt have food for to nourish thy life,
And God will provide for thy Children and wife.
Come all you forsaken, and be ruled by my own
If you will but listen to my moan;

8.3 Baseline 1 Poems

8.3.1 Poem 1

Keywords: "mask", "protective mask", "masks"

I will not go,
On the top of a glass-covered grave, I see
Who is good for us
And evil for me?

8.3.2 Poem 2

Keywords: "broccoli", "broccoli rabe", "broccolini"

The heart and soul
Are so well in harmony that we can go;
And yet the thought is folly
When thought of to have a good sense,

8.3.3 Poem 3

Keywords: "lacewing", "neuroptera", "trichoptera"

by the time of his speech.
He was able to see the face of the man,
And there was nothing for him to do;
Then he sailed away to the other side. His thoughts were as they

8.4 Baseline 2 Poems

8.4.1 Poem 1

Keywords: "church", "christian church", "churches"

From the Parson up in the Church;
And the Miser set up by a Priest:
All sorts of Books, yea, Gospel and Art
From Prose (in Age) and Fact

8.4.2 Poem 2

Keywords: "palace", "palaces", "royal palace"

Great Palaces so rare
that no Nation ever had,
and palace walls so brave
All of them with wonders wrought

8.4.3 Poem 3

Keywords: "mushroom", "mushrooms", "champignon"

Cherubs and Cherubins all
as with the Turtle-dove,
I do think it's better
Than the Picker-pigg of Westminster-Hall.

8.4.4 Poem 4

Keywords: "fountain", "fountains", "fount"

The Fountain and the Woods they so far did divide,
I saw one drowned in a grave, another in an amisse:
Both of them by this fountain one drop of blood was brought,
to spill in any number but two to