Outline

Introduction
What is Chinese poetry?
Poetic Rules
Goals
Our Approach

Planning
Word2vec
TextRank
Keyword Extraction & Expansion

Generation
Seq2Seq
Encoder: Bidirectional RNN

Decoder: Attention Mechanism
Alignment
Rhyming

Experimental Design
Training Data
Methods of Evaluation

Results
Evaluation
Turing Test

Future Improvements
RL Tuner
Beam Search
Convolutional Polishing
What is Chinese Poetry?

江雪
千山鸟飞绝，
万径人踪灭。
孤舟蓑笠翁，
独钓寒江雪。

River Snow
From hill to hill no bird in flight;
From path to path no man in sight.
A lonely fisherman afloat,
Is fishing snow in lonely boat.

Translated by Yuanchong Xu, http://localsev.lib.pku.edu.cn/bdms/mr_index.asp?id=57
Poetic Rules

Structure: Four lines, usually five or seven characters per line
Tone: P and Z each represents two tones
Rhyme: The last characters with ◎ must rhyme
Why study Chinese poetry?

Unique challenge - A lot of structure and pattern
Cultural importance - widely study today
Application in real life - teaching assistant
江雪

千山鸟飞绝，
万径人踪灭。

孤舟蓑笠翁，
独钓寒江雪。
**Goals**

<table>
<thead>
<tr>
<th>Thematic Correspondence:</th>
<th>Semantic Coherence:</th>
<th>Adherence to Poetic Rules:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Between sentences</td>
<td>Within sentences</td>
<td>Within poem</td>
</tr>
</tbody>
</table>

Can a deep neural network capture all of the **information** and **patterns**? Answer is **Yes** and **No**.
Our Approach

Planning Based Poetry Generation

Key Idea is separation of **Planning** & **Generation**

Thematic Correspondence  
Semantic Coherence

Planning
Word2vec

(Word Embedding)

Why do we need Word2Vec?
Word2vec

Word2vec uses a single hidden layer, fully connected neural network.

Source: https://www.tensorflow.org/tutorials/word2vec
Word2vec

Algorithm we use: Continuous Bag-of-Words model (CBOW)
The model predicts the current word from a window of surrounding context words

Source: https://www.tensorflow.org/tutorials/word2vec
Word2vec

Word2vec captures linguistic regularities - very important in our task.

Two interesting examples:

vec('Rome') = vec('Paris') – vec('France') + vec('Italy')

vec('Queen') = vec('King') – vec('man') + vec('woman')

Source: https://iksinc.wordpress.com/tag/continuous-bag-of-words-cbow/
TextRank

Algorithm:
1. Break sentences into segments.
2. Build weighted graph of segments
3. Run PageRank on graph (i.e. iterative ranking based with recommendation score of segment)
Keyword Extraction & Expansion

Input

Jieba Segmenter

Input Segments

Segment Ranking

Top Rank

Extracted Keywords

Top Similarity

Expanded Keywords

TextRank

Training Data

Jieba Segmenter

Training Segments

Segment-Level Word2vec

Top Rank

Top Similarity
Generating
Seq2seq

Common Applications:
- Machine Translation
- Question & Answering
- Text Generation

Decoder

Encoder

Bidirectional RNNs are based on the idea that the output at time $t$ may not only depend on the previous elements in the sequence, but also future elements.

Implementation: stack the forward and backward states and use them as input for decoder.

Source: https://www.semanticscholar.org/paper/A-Unified-Tagging-Solution-Bidirectional-LSTM-Recu-Wang-Qian/191dd7df9cb91ac22f56ed0dfa455651e8767a51
Encoder: Deep Bidirectional RNN

Similar to Bidirectional RNNs. Instead of single layer, have multiple layers per time step. Able to learn more complex behaviour.

Figure 1: Encoder-Decoder architecture with attention module. Section numbers reference experiments corresponding to the components.
Types of Attention

**Decoder State:**

\[ s_i = f(s_{i-1}, y_{i-1}, c_i) \]

**Context:**

\[ c_i = \sum_{j=1}^{T_x} \alpha_{i,j} h_j. \]

**Attention:**

\[ \alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T_x} \exp(e_{i,k})}, \]

\[ a_t(s) = \text{align}(h_t, \tilde{h}_s) = \frac{\exp(\text{score}(h_t, \tilde{h}_s))}{\sum_s \exp(\text{score}(h_t, \tilde{h}_s))} \]

**Bahdanau (Additive) Attention:**

Scoring function is neural network (single layer) applied on concatenation of encoder and decoder hidden states.

**Luong (Multiplicative) Attention:**

Generalizes the model and introduces new scoring functions:

\[ \text{score}(h_t, \tilde{h}_s) = \begin{cases} 
    h_t^\top \tilde{h}_s & \text{dot} \\
    h_t^\top W_a \tilde{h}_s & \text{general} \\
    v_a^\top W_a [h_t; \tilde{h}_s] & \text{concat} 
\end{cases} \]
Visualizing Attention
Loss Function

MSE
Due to the nature of MSE and Word2Vec, the output is not guaranteed to be a valid character. Its output is more like "feeling of a character". Based on our experiments (and eyeballing), the results are not as good.

Cross Entropy (maximize the log-likelihood)
Common loss function in similar tasks: text generation, machine translation, etc. Generated results look good, and this is the one we chose to use in some of our tests.
Rhyming: Heuristic

Inspiration: Poetry polishing
Poets usually polish their poetry

Realization: Word2vec
Word2vec model can find top N similar characters of a character
We can choose the one that rhymes

<table>
<thead>
<tr>
<th>Target Rhyme</th>
<th>Original</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>'间'</td>
<td>'山'</td>
<td>岩</td>
</tr>
<tr>
<td>Between</td>
<td>Mountain</td>
<td>Rock</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Rhyme</th>
<th>Original</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>'山'</td>
<td>'云'</td>
<td>烟</td>
</tr>
<tr>
<td>Mountain</td>
<td>Cloud</td>
<td>Smoke</td>
</tr>
</tbody>
</table>
Rhyming: Better than Heuristic

Before we disclose the secret, let’s take a look at the training data.

Source1: 飞
Target1: 千山鸟飞绝
Source2: 人 <PAD>
Target2: 万径人踪灭
Source3: 孤舟 <PAD>
Target3: 孤舟蓑笠翁
Source4: 雪 <PAD>
Target4: 独钓寒江雪

Keyword
Preceding Sentences
Surprise!
Reversing training data improves rhyming a lot.
Rhyming: Better than Heuristic

Why does reversing training data yields better rhyming?

Intuition:
RNN decides the last character first, then it is not subject to previously generated characters.
Alignment: Boosted Word2Vec

Idea:
Add vertical slices of poems as additional sentences in training word2vec model.

Goal:
Synthetically boost similarity between characters that appear in alignment in the training data.

Result:
Subtle change in order of words with top similarity
Positive effect by inspection
**Alignment:** Boosted Word2Vec

**Idea:**
Add vertical slices of poems as additional sentences in training word2vec model.

**Experiments:**

**Character:** 东
east

**Without Alignment:**
[西, 春, 隅, 南, 滨, 临]
[west, spring, corner, south, seaside, arrival]

**With Alignment:**
[西, 淮, 南, 江, 春, 北]
[west, river, south, river, spring, north]
Alignment: Aligning Training Data

Intuition:
Training data should be padded/aligned such that the location of keywords and each sentences are consistent
Experimental Design
Training Data

76,433 Poems
305,732 Lines
2,036,012 Characters
Methods of Evaluation

**BLEU Score:**
A score from 0 to 1 indicating how similar the candidate text is to the reference texts.

It is calculated on sentence level, but only the corpus level average is indicative of quality.

**Issue:**
Do not have good reference sentences

Not Yet Implemented

**Rhyming/Tonal Score:**
50% from rhyming,
50% from tonal.

- **Rhyming Score:**
  1: if end characters rhyme as expected
  0: otherwise

- **Tonal Score:**
  0 <= p <= 1: percentage of characters with expected tone types

**Structural Score:**
0: if lines are not five or seven characters, or have different lengths
1: otherwise

**Alignment Score:**
Train word2vec with only vertical slices of poems.
Use average similarity score across 4 sentences as poem alignment score

Not Yet Implemented
List of Training Params

Bidirectional: [True, False]
Decoder Input: [Ground Truth, Sampling]
Training Data Reverse: [True, False]
Training Data Alignment: [True, False]
Word2Vec Alignment: [True, False]
Cell Type: [LSTM, GRU]
Attention Type: [Bahdanau, Luong]
Hidden Units: 128
Depth: 4
Batch Size: 64
Results
Our Latest Model

Trained with:
Default setting
1,622,400 steps
~ 350 epochs
~ 4 days

Converged to:
Loss of 1.8

<table>
<thead>
<tr>
<th>Name</th>
<th>Smoothed</th>
<th>Value</th>
<th>Step</th>
<th>Time</th>
<th>Relative</th>
</tr>
</thead>
<tbody>
<tr>
<td>best/</td>
<td>1.823</td>
<td>1.810</td>
<td>1.622M</td>
<td>Sat Jul 22, 19:04:34</td>
<td>3d 22h 0m 34s</td>
</tr>
</tbody>
</table>
Input: 醉

Keywords:

酒
醒
醉
梅花

Poem:

舞困歌慵酒梦迟，
雪醒犹饮榻南池。
醉茶只说黄池主，
不看梅花便开时。

Generated Poem

Input: Drunk

Poem:

Sleepy dance, tired songs, and dream delayed by alcohol,
Awaken to ambrosia-like snow, lying in the south pond.
Drunk tea brought conversation about the golden pond,
Plum blossoms appear when none looks.
Rhyming/Tonal Score

Corresponds to 20.95% (10.68%) poems that do not rhyme.
Corresponds to 6.65% (0%) poems that have inconsistent lengths.
Turing Test

Designed a web app that lets users guess if the poetry sample was written by a person or a computer.

You can play the game here: http://ming-gpu-3.cs.uwaterloo.ca:8080
2500+ Data Points
From ~100 friends - a popular game!

43% Passed Turing Tests
Impressive given 39% of human poetry were labeled computer
### Turing Test: Breakdown

<table>
<thead>
<tr>
<th>Guessed \ Actual</th>
<th>RNN</th>
<th>HUMAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTER</td>
<td>695</td>
<td>525</td>
</tr>
<tr>
<td>HUMAN</td>
<td>491</td>
<td>774</td>
</tr>
</tbody>
</table>

- **User Clicked Computer:** 57%
- **User Clicked Human:** 37%
Rhymes very well
Beautiful words
Generally fluent
Coherent

Turing Test Insights

Weird length
Duplicate characters
“storyline”
Conflicting sentiment
Training Speed (4h)

Not using previous sentences vs
Not using bidirectional:
Doubles training speed, similar loss

Not using bidirectional vs Default:
Doubles training speed, significantly higher loss
Training Speed (24h)

**24 Hours of Training**

Not using previous sentences vs Not using bidirectional:
Doubles training speed, significantly higher loss

<table>
<thead>
<tr>
<th>Name</th>
<th>Smoothed</th>
<th>Value</th>
<th>Step</th>
<th>Time</th>
<th>Relative</th>
</tr>
</thead>
<tbody>
<tr>
<td>default/</td>
<td>2.521</td>
<td>2.539</td>
<td>211.5k</td>
<td>Sun Jul 23, 18:42:07</td>
<td>23h 32m 30s</td>
</tr>
<tr>
<td>no_bidirectional/</td>
<td>3.273</td>
<td>3.269</td>
<td>363.1k</td>
<td>Sun Jul 23, 18:42:55</td>
<td>23h 55m 3s</td>
</tr>
<tr>
<td>no_prev/</td>
<td>3.659</td>
<td>3.598</td>
<td>831.9k</td>
<td>Sun Jul 23, 18:42:29</td>
<td>23h 15m 13s</td>
</tr>
<tr>
<td>no_reverse_align/</td>
<td>2.865</td>
<td>2.875</td>
<td>217.9k</td>
<td>Sun Jul 23, 18:41:53</td>
<td>23h 49m 49s</td>
</tr>
</tbody>
</table>
## Training Stats (24h)

<table>
<thead>
<tr>
<th>Model/Stats</th>
<th>Epoch</th>
<th>Step</th>
<th>Perplexity</th>
<th>Loss</th>
<th>Sents/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>46</td>
<td>221500</td>
<td>12.08</td>
<td>2.521</td>
<td>127.76</td>
</tr>
<tr>
<td>No Reverse &amp; Alignment</td>
<td>47</td>
<td>228700</td>
<td>15.00</td>
<td>2.650</td>
<td>268.88</td>
</tr>
<tr>
<td>No Bidirectional</td>
<td>79</td>
<td>380500</td>
<td>24.86</td>
<td>3.234</td>
<td>288.93</td>
</tr>
<tr>
<td>No Previous</td>
<td>183</td>
<td>873800</td>
<td>37.55</td>
<td>3.546</td>
<td>896.23</td>
</tr>
</tbody>
</table>
Rhyming/Tonal Score

Observations:

**No Previous**
Penalized hard on sentence length

**No Bidirectional**
As good as default

**No Reverse & Alignment**
Penalized hard on rhyming
A Closer Look

Comparison of Score Distribution of Models

No Previous
Penalized hard on sentence length
A Closer Look

No Bidirectional
As good as default
A Closer Look

Comparison of Score Distribution of Models

No Reverse & Alignment
Penalized hard on rhyming
## Rhyming/Tonal Stats

<table>
<thead>
<tr>
<th>Model/Stats</th>
<th>Mean of Combined Score</th>
<th>Standard Deviation of Combined Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Data</td>
<td>0.8941</td>
<td>0.1843</td>
</tr>
<tr>
<td>Default</td>
<td>0.7802</td>
<td>0.2840</td>
</tr>
<tr>
<td>No Reverse &amp; Alignment</td>
<td>0.6997</td>
<td>0.2713</td>
</tr>
<tr>
<td>No Bidirectional</td>
<td>0.8025</td>
<td>0.2571</td>
</tr>
<tr>
<td>No Previous</td>
<td>0.1490</td>
<td>0.2346</td>
</tr>
</tbody>
</table>
Future Improvements
Integrate Heuristic with Model

During Training:
Convolutional Polishing

During Prediction:
Beam Search Optimization

Model Refinement After Training:
Reinforcement Learning Tuner
Goal:
Teach model structural/tonal/rhyming rules, while allowing it to learn patterns organically.

Key Idea:
Use trained model and poetry rules as reward to train a new reinforcement learning model.

To Approximate:
\[ Q(\text{state, action}) = \text{reward} \]
\[ \text{Likelihood given by trained model} \]
\[ \text{Score given by poetry rules} \]

Implementation of RL Tuner

Algorithm: Deep Double Q-Learning

\[ L_t(\theta_t) = (\log p(a|s) + \frac{1}{c} r_{MT}(a, s) + \gamma \max_{a'} Q(s', a'; \theta_{t-1}) - Q(s, a; \theta_t))^2 \]
Beam Search

A heuristic search algorithm that explores a graph by expanding the most promising node in a limited set.

- Computationally Efficient
- Able to Integrate Human Knowledge
- Able to consider the final performance

Source: https://en.wikipedia.org/wiki/Beam_search
Beam Search

Beam search uses **BFS** to build its search tree.

At each level of the tree, it generates **all successors** of the states at the current level, **sorting** them in increasing order of **heuristic cost** (possibly domain knowledge!)

However, it only stores a predetermined number, $\beta$, of best states at each level.
Polishing Network

**Inspiration:**

Human poets often draft and **recompose** clauses numerous times before settling for the best formulation.

It’s an **iterative process**, where output from a previous generation informs the next generation.
Convolutional Polishing

**Improved Formulation:**
Integrate polishing network with decoder, instead of using it as an output layer.

**Why Convolutional?**
Fixed sized windows helps to extract local (neighboring) patterns of successive characters.

\[
\begin{align*}
    h_i^{(n+1)} &= f(W_x x_{i-1} + W_h h_i^{(n+1)}) \\
    &+ \text{Attention} + \text{Polish}
\end{align*}
\]
Convolutional Polishing

When to Stop?
- When change made by polishing is small enough (e.g. cosine similarity of encoded).
- Polishing may not converge, need termination threshold.

Issues:
- Complex architecture, hard to implement.
- Long training time with large number of iterations per sample.
Thanks!
Any questions?
References: Papers

**Scheduled Sampling**
Title: Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks

**Beam Search**
Title: Sequence-to-Sequence Learning as Beam-Search Optimization

**RL Tuner**
Title: Tuning Recurrent Neural Networks with Reinforcement Learning
Link: [https://arxiv.org/pdf/1611.02796v2.pdf](https://arxiv.org/pdf/1611.02796v2.pdf)
Source: [https://github.com/tensorflow/magenta/tree/master/magenta/models/rl_tuner](https://github.com/tensorflow/magenta/tree/master/magenta/models/rl_tuner)

Title: Deep Reinforcement Learning for Dialogue Generation
Note: Augmenting seq2seq with reinforcement learning
References: Source Code

JayParks/tf-seq2seq
Link: [https://github.com/JayParks/tf-seq2seq](https://github.com/JayParks/tf-seq2seq)
Description:
- RNN encoder-decoder architectures and attention mechanism
- Implemented using the latest (1.2) tf.contrib.seq2seq modules
Usage: consulted architecture code snippet

DevinZ1993/Chinese-Poetry-Generation
Description:
- An undergraduate student’s attempt to implement planning based poetry generation
- Produce good but not excellent results
Usage: consulted data utility code snippet
References: Source Code

tensorflow/tensorflow/contrib/seq2seq/
Link: [https://github.com/tensorflow/tensorflow/tree/r1.2/tensorflow/contrib/seq2seq](https://github.com/tensorflow/tensorflow/tree/r1.2/tensorflow/contrib/seq2seq)
Docs: [https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq](https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq)
Description:
- Officially endorsed components used for implementing sequence to sequence translation networks
- **Caveat:** Volatile API especially prior to Tensorflow 1.2 release. Does not correspond to some of the seq2seq tutorials on the Tensorflow documentation/tutorial site (which uses a legacy version of the framework)
Usage: used as main building block of current implementation

farizrahman4u/seq2seq
Link: [https://github.com/farizrahman4u/seq2seq](https://github.com/farizrahman4u/seq2seq)
Description:
- A Keras seq2seq framework implementing attention, bidirectional encoder
- **Caveat:** Large number of issues tracked on GitHub. We failed to get this working. Training loss is consistently high after many epochs, and only gibberish was generated.
Usage: Failed to get this working
Acknowledgement

Dr. Ming Li - University of Waterloo

Dr. Xiaopeng Yang - University of Waterloo