Chinese Poetry Generation

Simon & Vera

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	River Snow
千山鸟飞绝,	From hill to hill no bird in flight;
万径人踪灭。	From path to path no man in sight.
孤舟蓑笠翁,	A lonely fisherman afloat,
独钓寒江雪。	Is fishing snow in lonely boat.

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

Poetic Rules

(P) P (Z) Z Z P P, ◎
(Z) Z P P Z Z P。◎
(Z) Z (P) P P Z Z ,
(P) P (Z) Z Z P P。◎

(P) P (Z) Z P P Z,
(Z) Z P P Z Z P。◎
(Z) Z (P) P P Z Z,
(P) P (Z) Z Z P P。◎

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

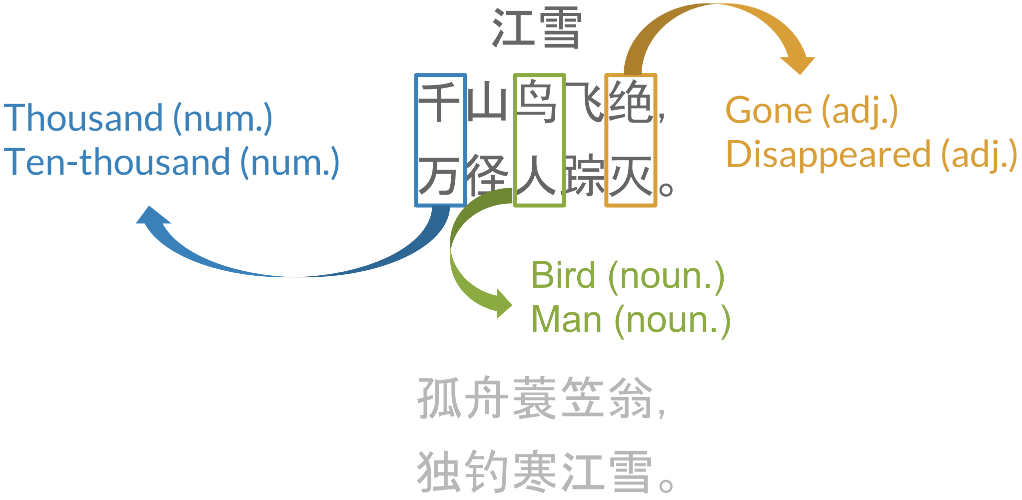
Why study Chinese poetry?

Unique challenge - A lot of structure and pattern

Cultural importance - widely study today

Application in real life - teaching assistant

Character Alignment



Goals

Thematic Correspondence:

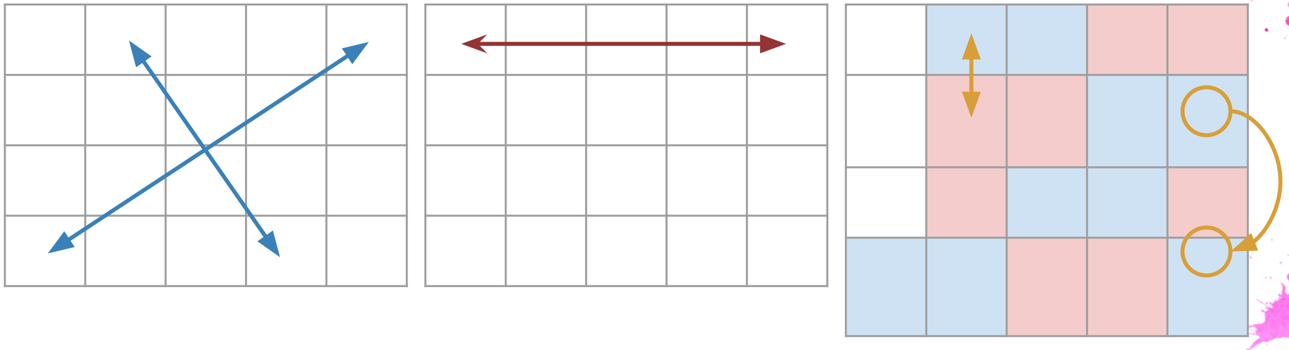
Between sentences

Semantic Coherence:

Within sentences

Adherence to Poetic Rules:

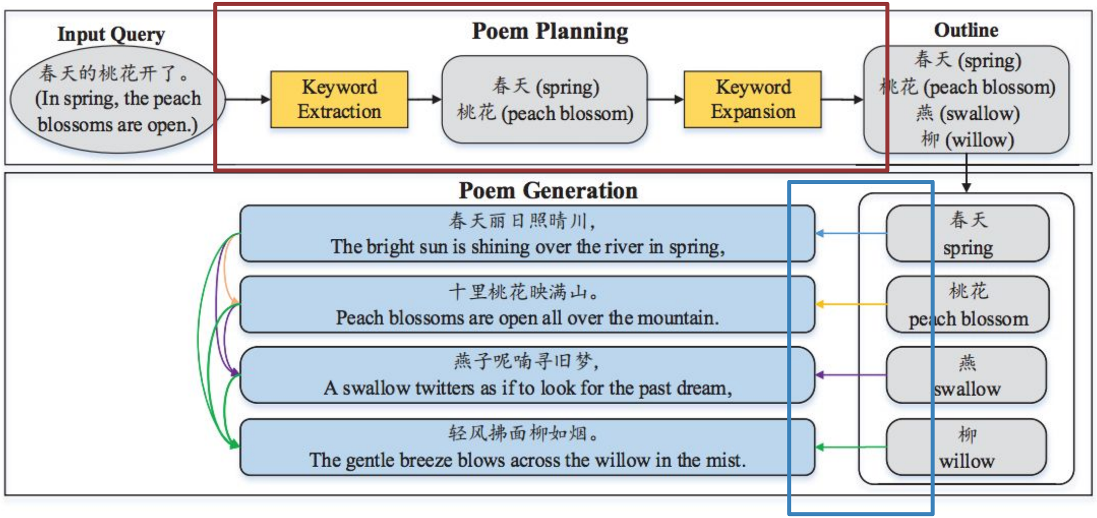
Within poem



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**

Source: https://arxiv.org/pdf/1610.09889.pdf

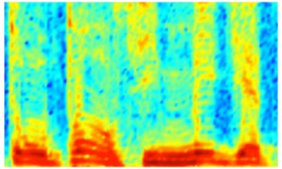
Planning

Word2vec
(Word Embedding)

Why do we need Word2Vec?

Word2vec

AUDIO



Audio Spectrogram

DENSE

IMAGES

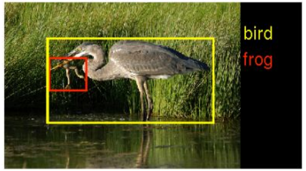


Image pixels

DENSE

TEXT

0	0	0	0.2	0	0.7	0	0	0	...	...

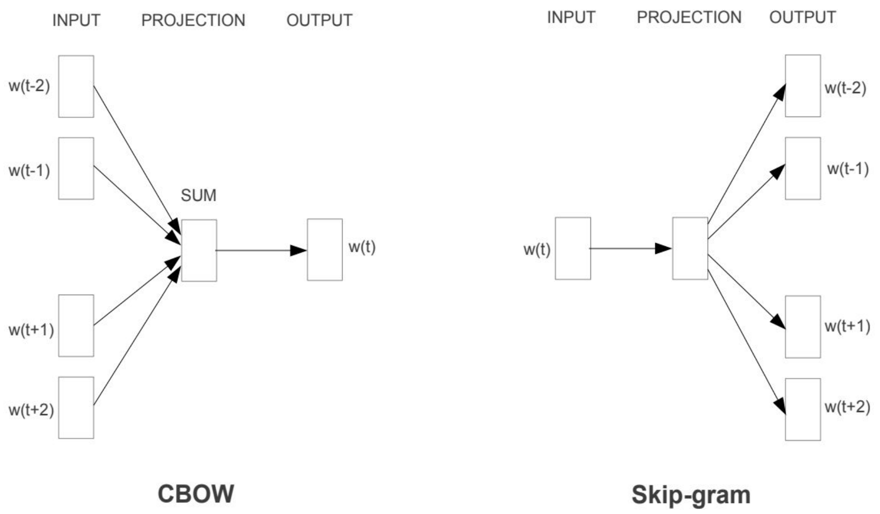
Word, context, or document vectors

SPARSE

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
https://arxiv.org/pdf/1301.3781.pdf

Word2vec

Word2vec captures linguistic regularities - very important in our task.

Two interesting examples:

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

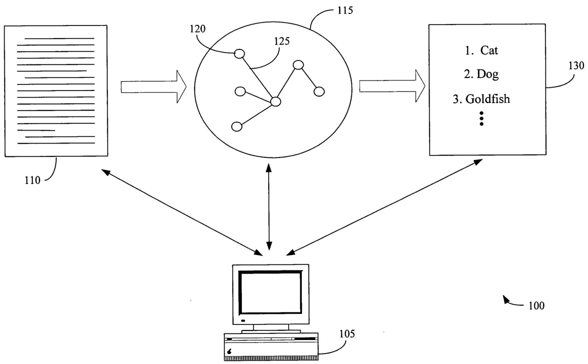
vec('Queen') = vec('King') – vec('man') + vec('woman')

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

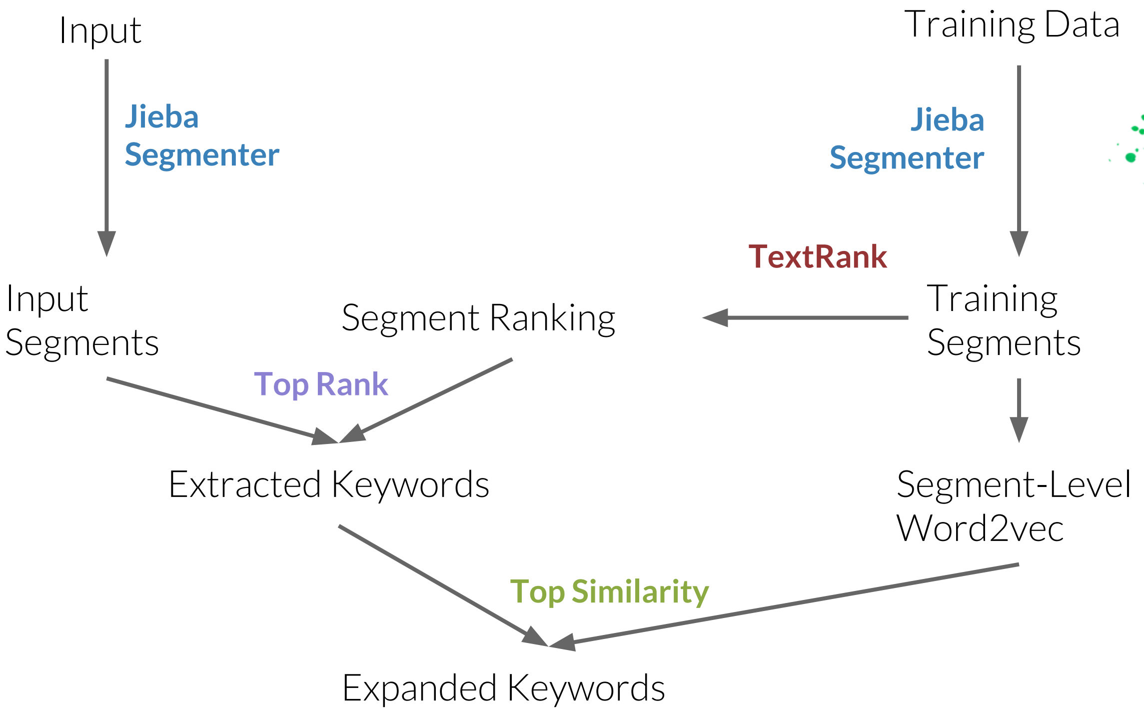
TextRank

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)

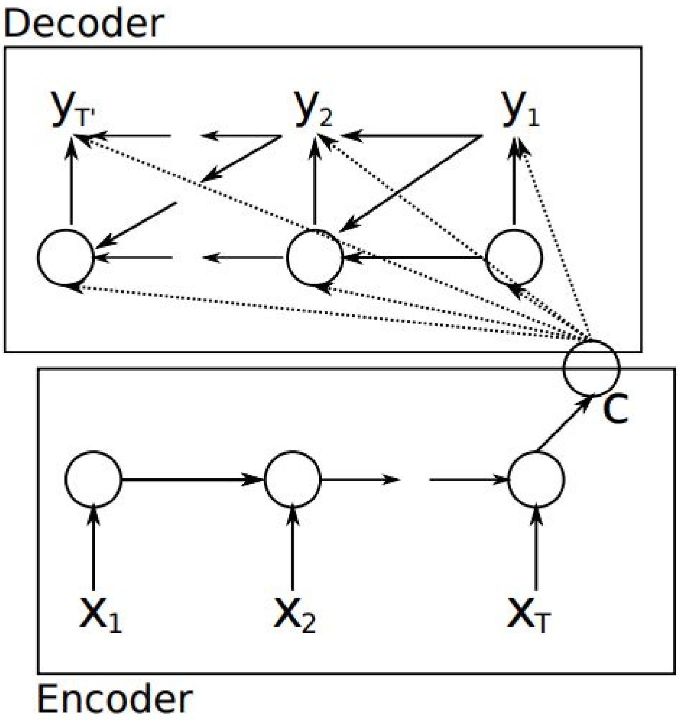


Keyword Extraction & Expansion



Generating

Seq2seq

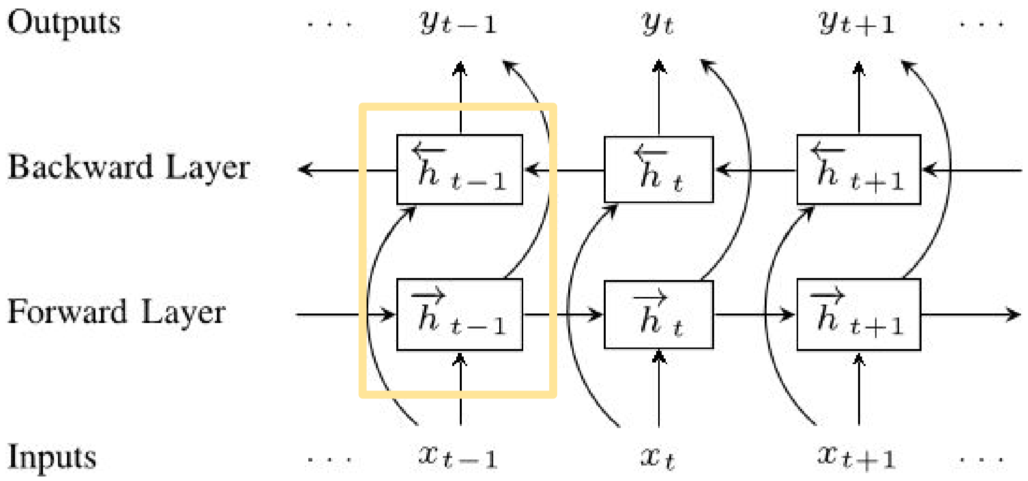


Common Applications:
Machine Translation
Question & Answering
Text Generation

Source: https://arxiv.org/pdf/1406.1078.pdf

Encoder: Bidirectional RNN

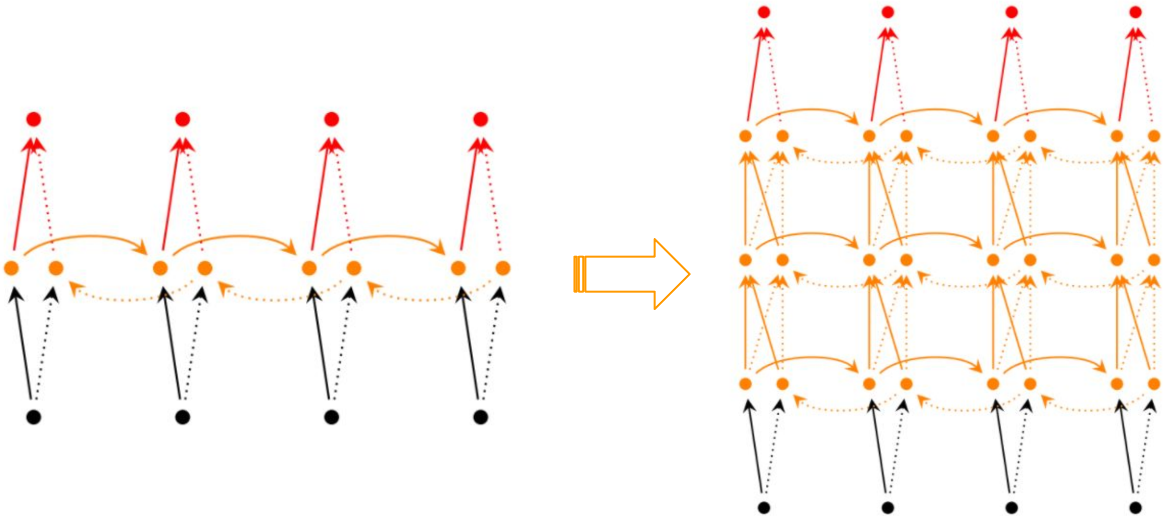
Bidirectional RNNs are based on the idea that the output at time t may not only depend on the previous elements in the sequence, but also future elements.



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/191dd7df9cb91ac22f56ed0dfa4a5651e8767a51

Encoder: Deep Bidirectional RNN



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

Source: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Attention Decoder

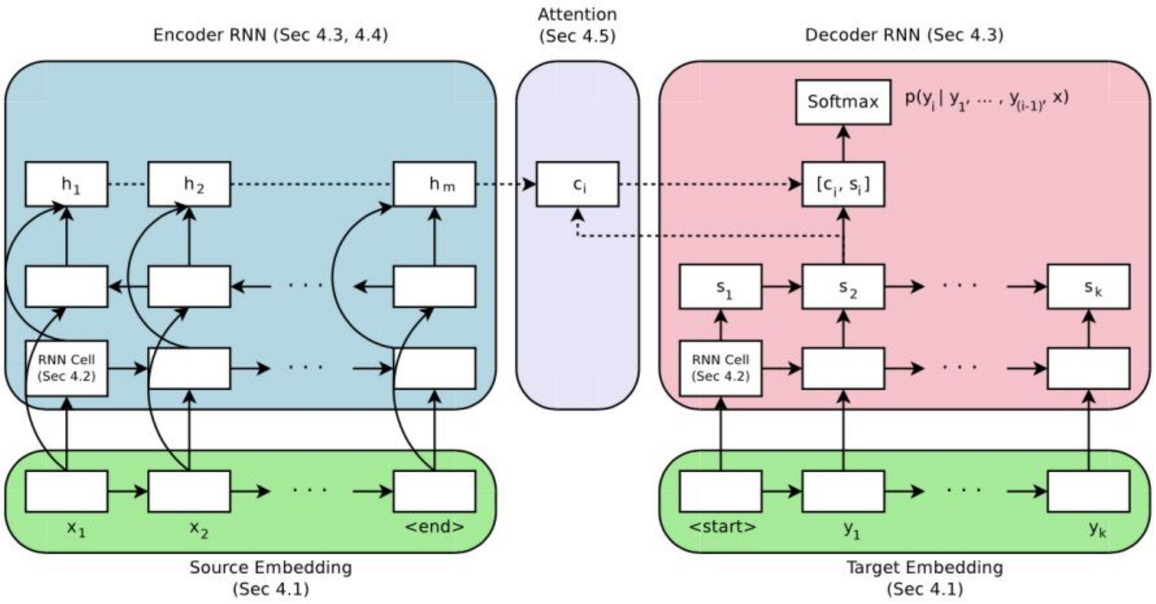


Figure 1: Encoder-Decoder architecture with attention module. Section numbers reference experiments corresponding to the components.

Types of Attention

Decoder State:

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

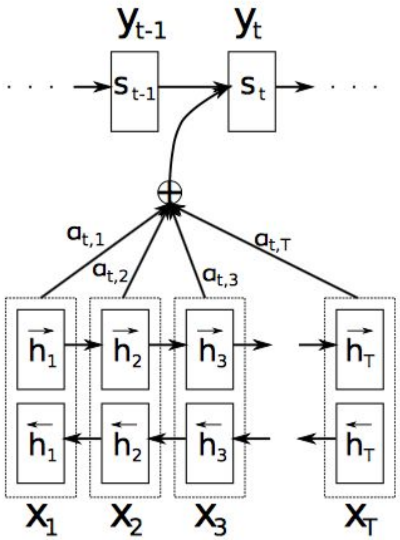
Context:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

Attention:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

$$a_t(s) = align(h_t, \bar{h}_s) = \frac{exp(score(h_t, \bar{h}_s))}{\sum_{s'} exp(score(h_t, \bar{h}_{s'}))}$$



Bahdanau (Additive) Attention:

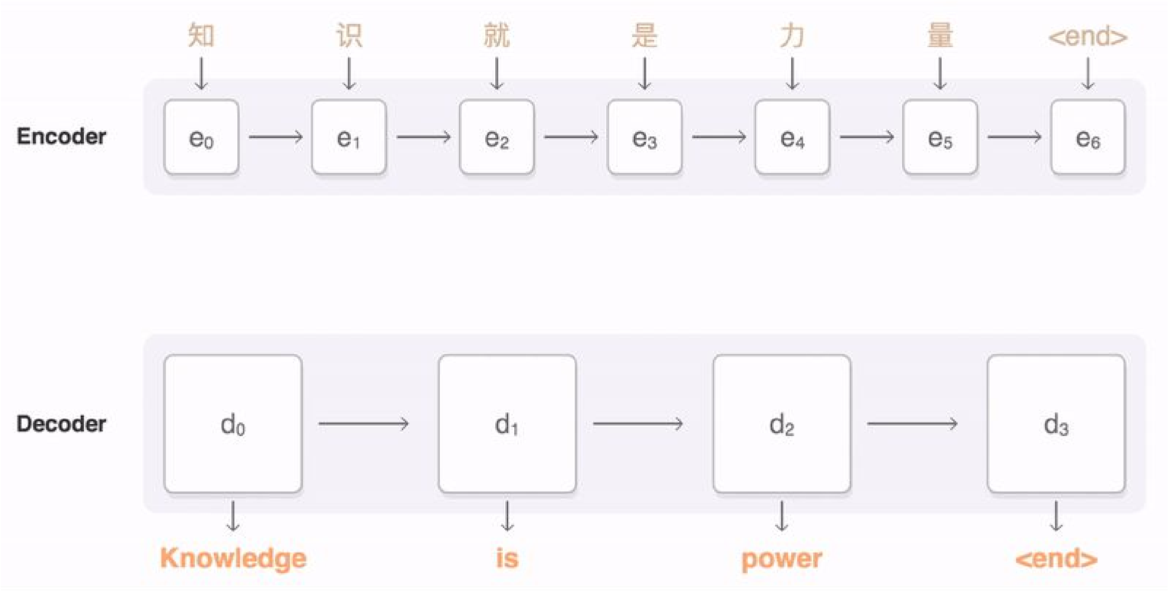
Scoring function is neural network (single layer) applied on concatenation of encoder and decoder hidden states.

Luong (Multiplicative) Attention:

Generalizes the model and introduces new scoring functions:

$$score(h_t, \bar{h}_s) = \begin{cases} h_t^\top \bar{h}_s & dot \\ h_t^\top \mathbf{W}_a \bar{h}_s & general \\ v_a^\top \mathbf{W}_a [h_t; \bar{h}_s] & concat \end{cases}$$

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

Target Rhyme Original Output

refine ('间', '山') = 岩

Between Mountain Rock

Target Rhyme Original Output

refine ('山', '云') = 烟

Mountain Cloud Smoke

Rhyming: Better than Heuristic

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

千山鸟飞绝 飞: Fly
万径人踪灭 人: Person
孤舟蓑笠翁 孤舟: Lonely Boat
独钓寒江雪 雪: Snow

Keyword

Preceding Sentences

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

Surprise!

Reversing training data improves rhyming a lot.

Rhyming: Better than Heuristic

千山鸟飞绝	飞: Fly
万径人踪灭	人: Person
孤舟蓑笠翁	孤舟: Lonely Boat
独钓寒江雪	雪: Snow

Source1:	飞
Target1:	千山鸟飞绝
Source1:	飞
Target1:	绝飞鸟山千

Source2:	人<PAD>千山鸟飞绝
Target2:	万径人踪灭
Source2:	绝飞鸟山千<PAD>人
Target2:	灭踪人径万

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.

江雪

千山鸟飞绝，
万径人踪灭。
孤舟蓑笠翁，
独钓寒江雪。

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.

江雪

千山鸟飞绝，
万径人踪灭。
孤舟蓑笠翁，
独钓寒江雪。

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

K	W	S	L	I	N	E	1	S	L	I	N	E	2	S

K	W	*	*	S	L	I	N	E	1	*	*	S	L	I	N	E	2	*	*	S

K	W	P	P	S	L	I	N	E	1	P	P	S	L	I	N	E	2	P	P	S

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

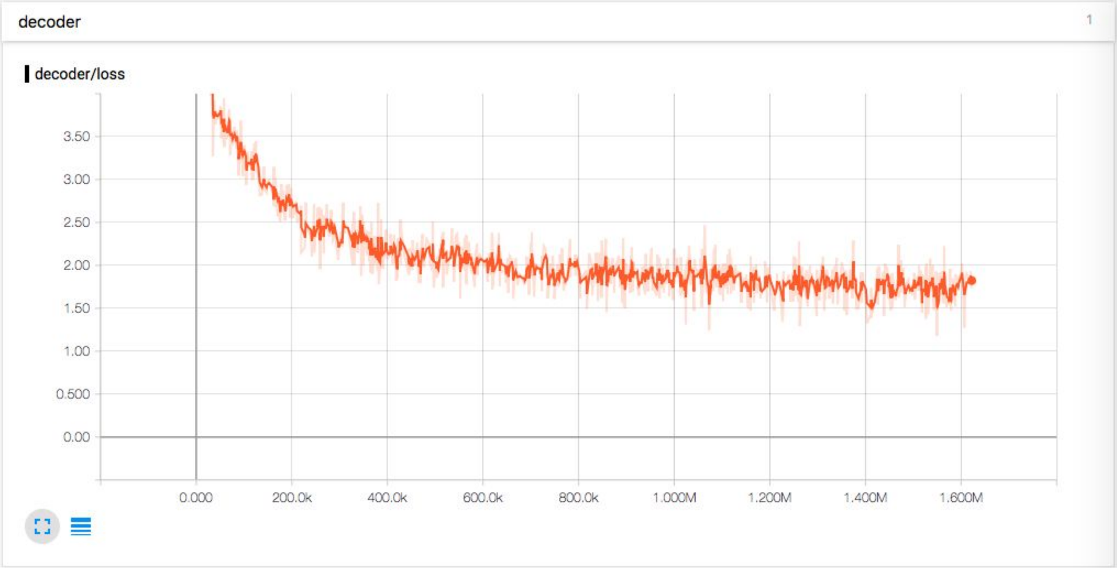
List of Training Params

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

[Default, Alternative]

Results

Our Latest Model



	Name	Smoothed	Value	Step	Time	Relative
	best/.	1.823	1.810	1.622M	Sat Jul 22, 19:04:34	3d 22h 0m 34s

Trained with:

Default setting

1,622,400 steps

~ **350** epochs

~ **4** days

Converged to:

Loss of **1.8**

Generated Poem

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。

Input: Drunk

Poem:

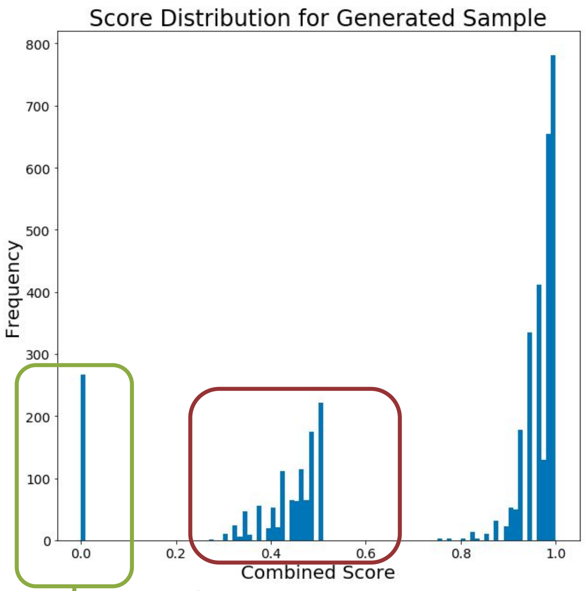
Sleepy dance, tired songs, and dream delayed by **alcohol**,

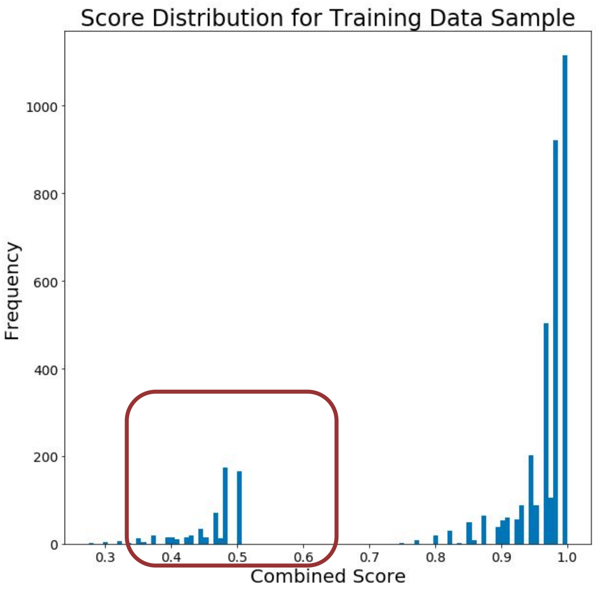
Awaken to ambrosia-like snow, lying in the south pond.

Drunk tea brought conversation about the golden pond,

Plum blossoms appear when none looks.

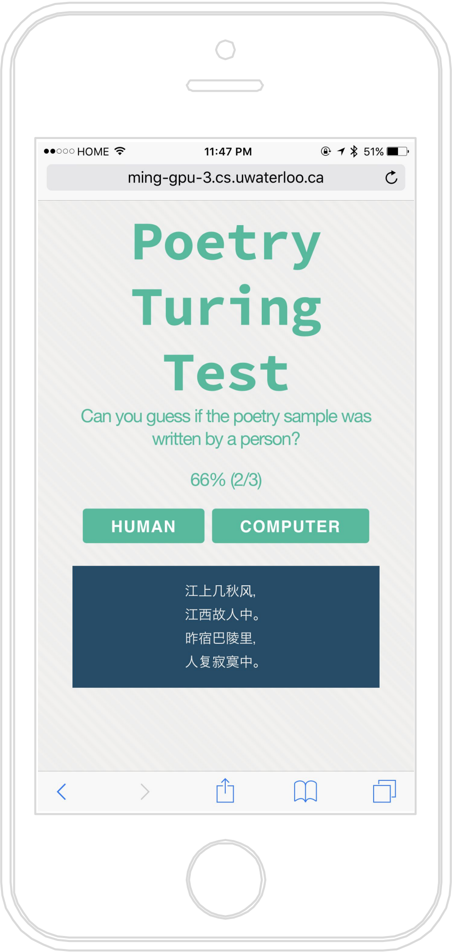
Rhyming/Tonal Score





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

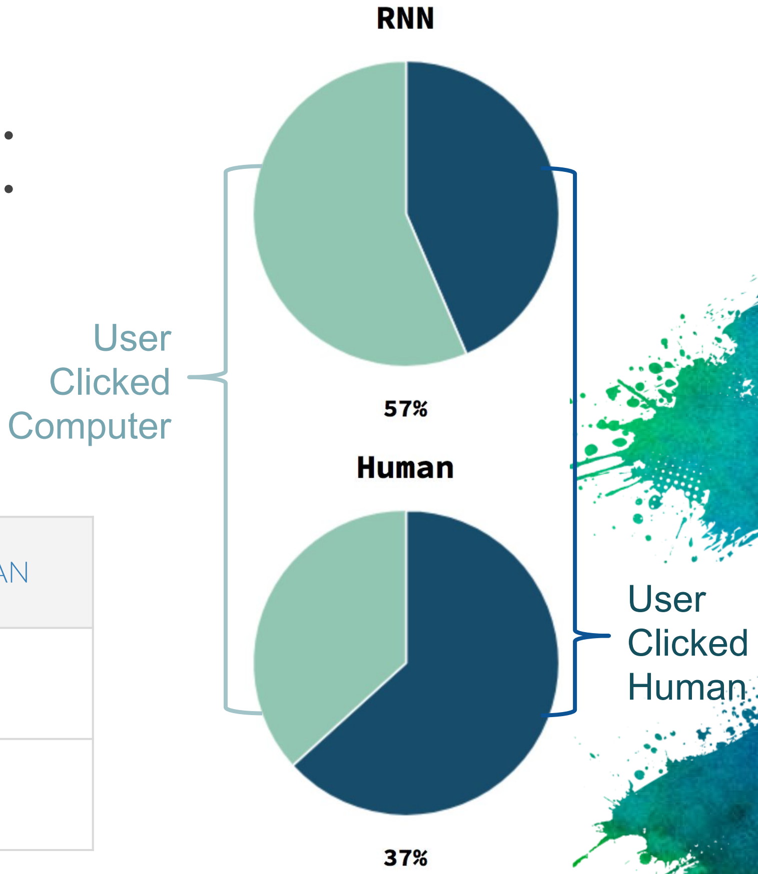
From ~100 friends - a popular game!

43% Passed Turing Tests

Impressive given 39% of human poetry were labeled computer

Turing Test: Breakdown

Guessed \ Actual	RNN	HUMAN
COMPUTER	695	**525**
HUMAN	**491**	774



Turing Test

Insights

Rhymes very well

Beautiful words

Generally fluent

Coherent

...

Weird length

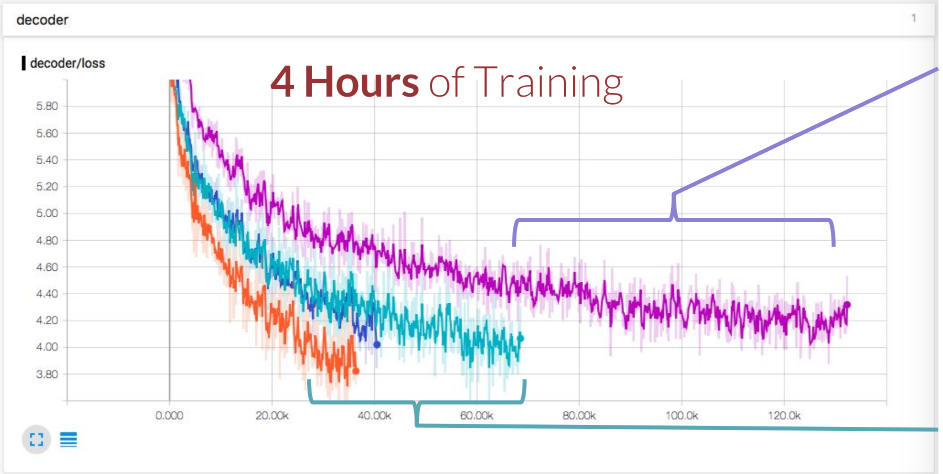
Duplicate characters

“storyline”

Conflicting sentiment

...

Training Speed (4h)

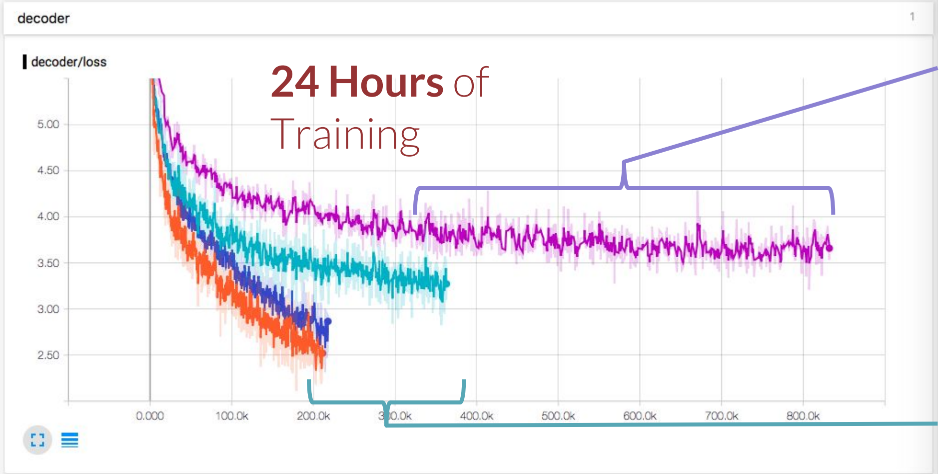


	Name	Smoothed	Value	Step	Time	Relative
	default/.	3.779	3.650	37.00k	Sat Jul 22, 23:11:06	4h 3m 4s
	no_bidirectional/.	4.122	4.268	70.20k	Sat Jul 22, 23:12:37	4h 25m 57s
	no_prev/.	4.117	4.085	135.9k	Sat Jul 22, 23:12:38	3h 46m 42s
	no_reverse_align/.	4.081	4.002	41.30k	Sat Jul 22, 23:12:24	4h 21m 52s

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)



Name	Smoothed	Value	Step	Time	Relative
default/.	2.521	2.539	211.5k	Sun Jul 23, 18:42:07	23h 32m 30s
no_bidirectional/.	3.273	3.269	363.1k	Sun Jul 23, 18:42:55	23h 55m 3s
no_prev/.	3.659	3.598	831.9k	Sun Jul 23, 18:42:29	23h 15m 13s
no_reverse_align/.	2.866	2.875	217.9k	Sun Jul 23, 18:41:53	23h 49m 49s

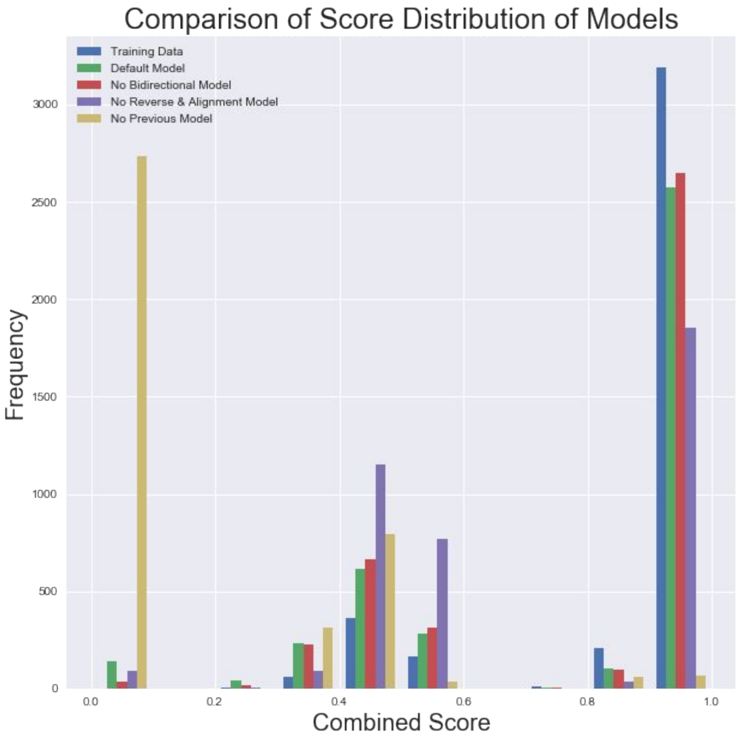
Not using previous sentences vs **Not using bidirectional:**
Doubles training speed, significantly higher loss

Not using bidirectional vs **Default:**
Doubles training speed, significantly higher loss

Training Stats (24h)

Model/Stats	Epoch	Step	Perplexity	Loss	Sents/s
Default	46	221500	12.08	2.521	127.76
No Reverse & Alignment	47	228700	15.00	2.650	268.88
No Bidirectional	79	380500	24.86	3.234	288.93
No Previous	183	873800	37.55	3.546	896.23

Rhyming/Tonal Score



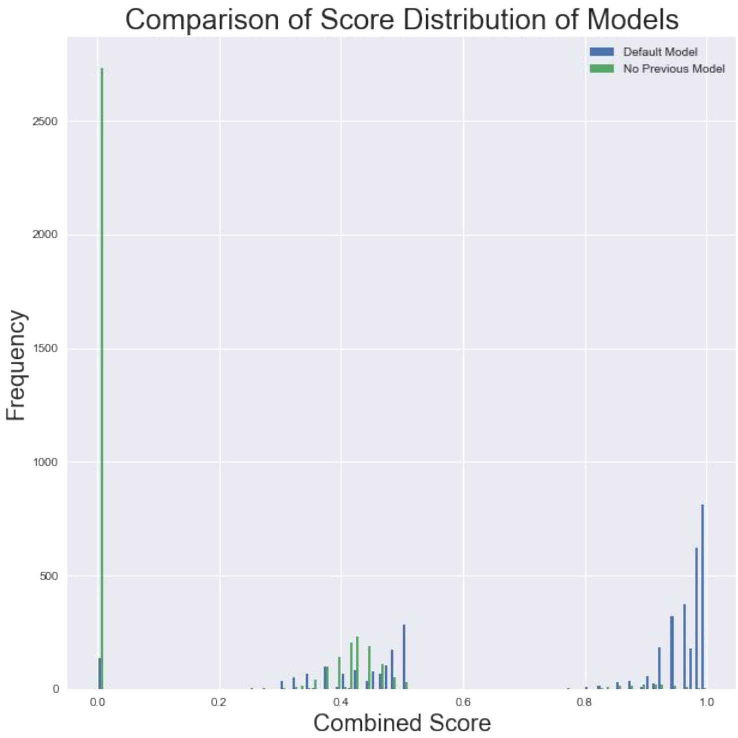
Observations:

No Previous
Penalized hard on sentence length

No Bidirectional
As good as default

No Reverse & Alignment
Penalized hard on rhyming

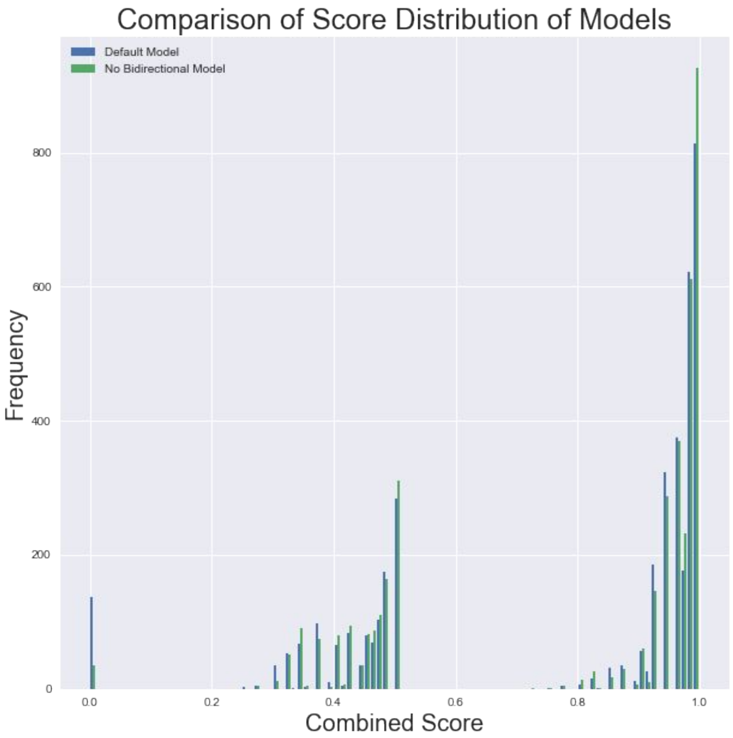
A Closer Look



No Previous

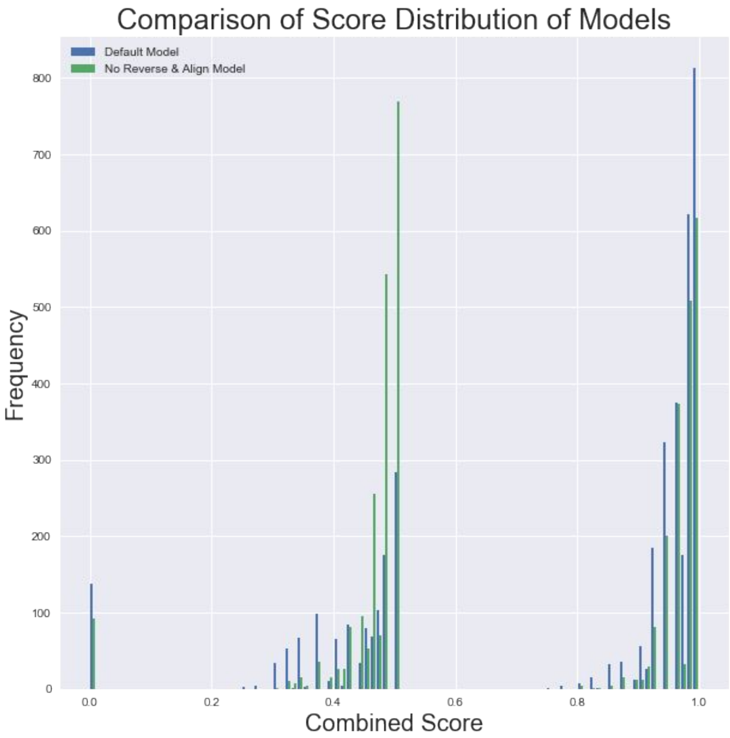
Penalized hard on sentence length

A Closer Look



No Bidirectional
As good as default

A Closer Look



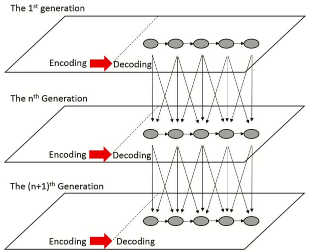
No Reverse & Alignment
Penalized hard on rhyming

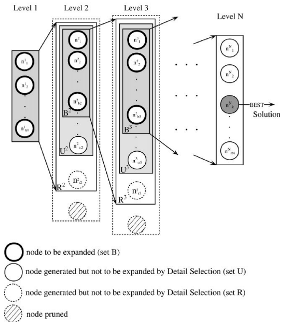
Rhyming/Tonal Stats

Model/Stats	Mean of Combined Score	Standard Deviation of Combined Score
Training Data	0.8941	0.1843
Default	0.7802	0.2840
No Reverse & Alignment	0.6997	0.2713
No Bidirectional	0.8025	0.2571
No Previous	0.1490	0.2346

Future Improvements

During Training:
Convolutional Polishing

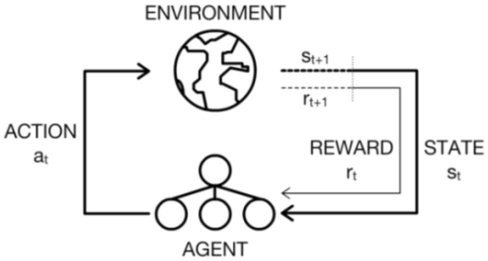




During Prediction:
Beam Search Optimization

Integrate Heuristic with Model

Model Refinement After Training:
Reinforcement Learning Tuner



RL 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.

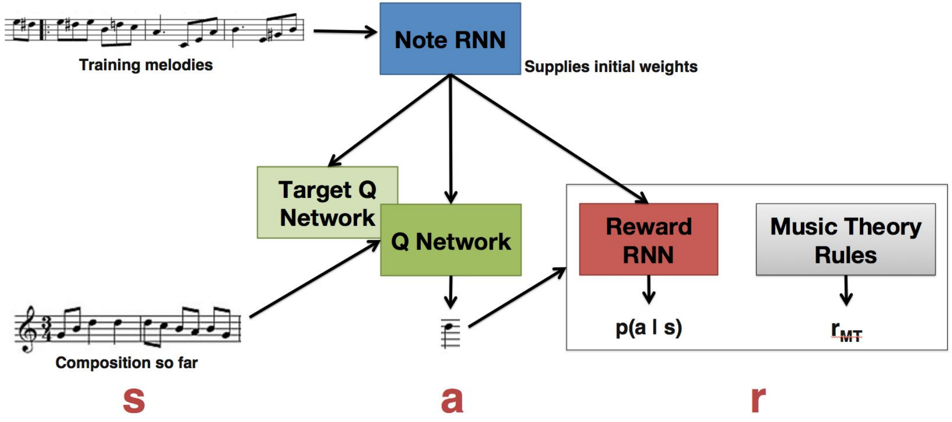
To Approximate:

Q(**state**, **action**) = **reward** $\begin{cases} \text{Likelihood given by trained model} \\ \text{Score given by poetry rules} \end{cases}$

Source: https://magenta.tensorflow.org/2016/11/09/tuning-recurrent-networks-with-reinforcement-learning

Implementation of RL Tuner

Algorithm: Deep Double Q-Learning



Likelihood — Score — Discounted Future Return — Predicted Reward

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

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

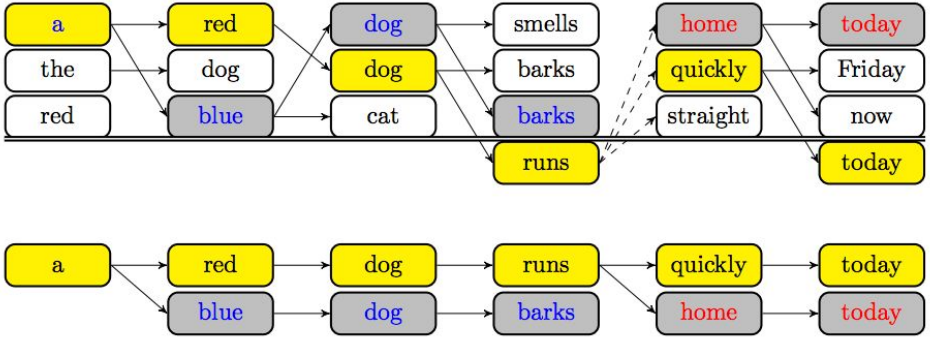
Source: https://en.wikipedia.org/wiki/Beam_search
https://arxiv.org/pdf/1606.02960.pdf

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, β, of best states at each level.

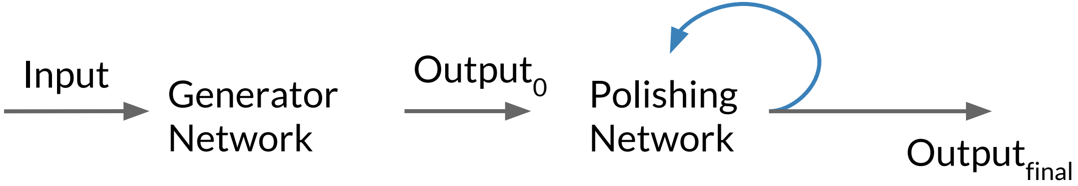


Polishing Network

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.



Source: https://pdfs.semanticscholar.org/0bd8/5c60a9ebcee8072245ebe499c8e3b26651cf.pdf

Convolutional Polishing

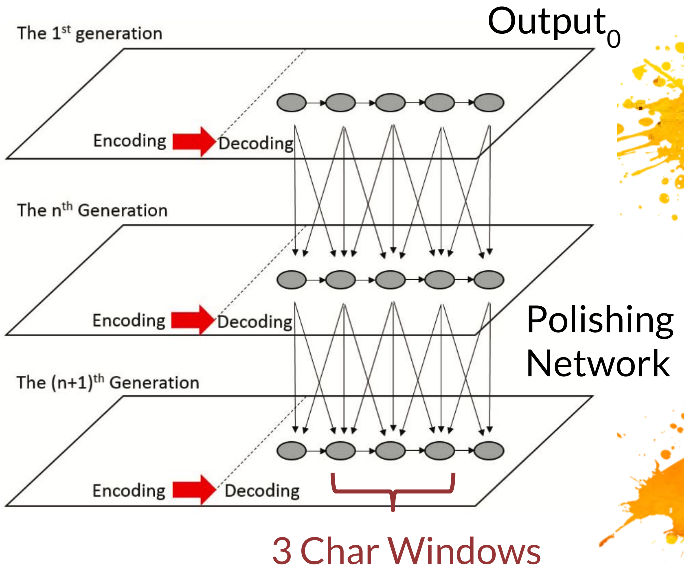
Improved Formulation:

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.

$$\boldsymbol{h}_i^{(n+1)} = f(\overbrace{W_x \boldsymbol{x}_{i-1}}^{\text{Input}} + \overbrace{W_h \boldsymbol{h}_{i-1}^{(n+1)}}^{\text{State}} + \underbrace{W_c \boldsymbol{c}_i^{(n+1)}}_{\text{Attention}} + \underbrace{W_o \boldsymbol{o}_i^{(n)}}_{\text{Polish}})$$



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
Link: [https://arxiv.org/abs/1506.03099]

Beam Search
Title: Sequence-to-Sequence Learning as Beam-Search Optimization
Link: [https://arxiv.org/abs/1606.02960]

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

Title: Deep Reinforcement Learning for Dialogue Generation
Link: [https://arxiv.org/pdf/1606.01541.pdf]
Note: Augmenting seq2seq with reinforcement learning

References:Source Code

JayParks/tf-seq2seq
Link: [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
Link: [https://github.com/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
Docs: [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]
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