Objective

- Two-sentence horror stories are a form of flash fiction in which a self-contained horror story is told in only two sentences.
- Goal: Given the first sentence, generate the second sentence of a two-sentence horror story.
- Challenges: distinctive style, need for a full narrative arc in a limited scope.
- Evaluation targets: language fluency and story cohesiveness.

Dataset

- Stories scraped from subreddit /r/TwoSentenceHorror.
- Preprocessing:
  - Replaced out-of-vocabulary characters with ASCII equivalents.
  - Filtered out non-story posts and incorrect-length posts.
  - Split stories into first and second sentences.
- Generation dataset: 22,000 examples total.
- Classification dataset: for evaluating story cohesiveness: 44,000 examples.
  - Shuffled second sentences between examples.
  - 22,000 original ex. = "cohesive", 22,000 shuffled ex. = "not cohesive".
  - 85% training, 7.5% validation, 7.5% test for both datasets.

Baseline

- Non-neural retrieval baseline using Jaccard similarity.
  - Jaccard = \frac{|A \cap B|}{|A \cup B|}.
  - Ratio of number of unique words common to both texts to number of unique words across both texts.
  - Useful indicator of contextual similarity.
- Generation: For each example, find and output target sentence from training dataset with highest Jaccard similarity to input text.

Model Architectures

- **Transformer**
  - Based on Vaswani’s transformer model [1].
  - 6-layer encoders/decoders, 512 hidden units each.
  - 8 attention heads, scaled dot-product self-attention.
  - Feed-forward network layer w/ 2048 units, dropout 0.1.
  - 512-d embeddings + sinusoidal position encoding.
  - trained with Adam optimizer and NLL loss.

- **LSTM**
  - 2-layer unidirectional LSTM encoder and decoder with multiplicative attention.
  - encoder & decoder have 500 hidden units.
  - 500-d word embeddings, dropout 0.3.
  - trained with SGD and NLL loss.

- **BLSTM**
  - Same as LSTM, but uses a bidirectional encoder.

Decoding Algorithms

- **Random top-k sampling** for k = 3, 5, 8, 10.
  - Temperature = 0.5, 0.75, 1.0.
  - Higher temperature led to more diversity, but also less readable output.
- **Beam search** with beam size = 3, 5, 8, 10.
  - Higher beam size yielded better output that was more readable and internally consistent.
  - N-gram blocking was not useful.

Evaluation

- All models deliberately overfit to act as retrieval models.
- Finding a stopping point for overfitting:
  - N-gram similarity - % n-grams in gen. output that exist in train.
  - Exact match - # gen. outputs that exactly match a sentence in train.
- Facebook’s InferSent [2]:
  - Used for automatic cohesion eval.
  - BLSTM sentence encoder with max-pooling + classifier.
  - Encoder trained on GloVe Wiki 2014 300-dim word embeddings.
  - Pre-trained on SNLI, then trained on classification dataset using SGD.
  - Achieved 78.4/70.1% train/test acc.
- Human evaluation (25-question survey):
  - 5 samples/model, randomly selected.
  - Questions on readability, cohesion, "horror", Ai detection (rated 1-10).

Results

<table>
<thead>
<tr>
<th>Automated Metric</th>
<th>2-gram Sim. %</th>
<th>4-gram Sim. %</th>
<th>6-gram Sim. %</th>
<th># Exact Match</th>
<th>Avg. Jaccard</th>
<th>% Cohesive</th>
<th>Avg. Readability</th>
<th>Avg. Cohesion</th>
<th>Avg. Quality</th>
<th>% AI vs Human Acc.</th>
<th>Survey Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>78.92</td>
<td>25.53</td>
<td>3.22</td>
<td>0.068</td>
<td>78.76</td>
<td></td>
<td>8.54</td>
<td>8.59</td>
<td>7.69</td>
<td>84.44</td>
<td>8.75</td>
</tr>
<tr>
<td>Jaccard</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>1864</td>
<td>0.271</td>
<td>85.78</td>
<td>7.14</td>
<td>5.73</td>
<td>5.71</td>
<td>61.48</td>
<td>7.46</td>
</tr>
<tr>
<td>LSTM</td>
<td>94.12</td>
<td>42.28</td>
<td>16.13</td>
<td>0.032</td>
<td>81.16</td>
<td></td>
<td>6.74</td>
<td>5.21</td>
<td>5.28</td>
<td>68.15</td>
<td>6.63</td>
</tr>
<tr>
<td>BLSTM</td>
<td>95.03</td>
<td>46.46</td>
<td>17.46</td>
<td>0.055</td>
<td>61.00</td>
<td></td>
<td>6.63</td>
<td>4.43</td>
<td>4.63</td>
<td>81.48</td>
<td>6.33</td>
</tr>
<tr>
<td>Transformer</td>
<td>99.68</td>
<td>92.94</td>
<td>89.82</td>
<td>0.046</td>
<td>47.16</td>
<td></td>
<td>6.93</td>
<td>5.11</td>
<td>5.06</td>
<td>68.88</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Tokenized entries included for completeness, but not very meaningful.

Table 2: Aggregated results from 27 respondents.

References


Sample Stories

- **Human sample:** My neighbor’s dog dug up some human bones. Guess I gotta start burying them deeper.
- **Transformer sample:** I had a staring contest with my reflection. I won.
- **LSTM sample:** I really hate when I burn dinner. Thankfully the neighbors have a new dog.
- **Most mistaken as human-generated (actually Jaccard):** What can i say, when my life has been a series of farewells, to my family, friends, and all i hold dear? This is all i can remember of the last moments of my life and i’m damned to re-live them over and over and over.

Discussion

- Transformer was the best retrieval model and produced readable but not very cohesive output.
- Jaccard sim. produced surprisingly cohesive stories.
- Using NMT-style seq2seq approach generally effective.
- Output difficult to distinguish from human stories.
- Story cohesion is difficult to measure.

Can you tell the difference between AI-generated and human-generated stories? Take our survey: bit.ly/horrifAI