Make RNNs Great Again
Character-Level Sequence Generation in the Style of Donald Trump

Benjamin Sherman
&
Zayd Hammoudeh

Dept. of Computer Science
University of California, Santa Cruz

December 5, 2017
Quiz

Sherman & Hammoudeh

Introduction

Learner Architecture

Training Overview

Training Overview

Text Generation

Decision Engine

Conclusions

Dept. of Computer Science
UC, Santa Cruz

Two Quotes: One computer-generated and one from Donald Trump. Can you identify the real one?

▶ Quote #1: You look at the nuclear deal, thing that real really bothers me, it would have been so easy and its not - as important as these lives are - nuclear is so powerful. My uncle explained that to me many, many years ago...
**Two Quotes:** One computer-generated and one from Donald Trump. Can you identify the real one?
Two Quotes: One computer-generated and one from Donald Trump. Can you identify the real one?

► Quote #1: *You look at the nuclear deal*, thing that real really bothers me, it would have been so easy and its not - as important as these lives are - nuclear is so powerful. My uncle explained that to me many, many years ago...
Quiz

Two Quotes: One computer-generated and one from Donald Trump. Can you identify the real one?

► Quote #2: You look at the nuclear deal, and it’s going to be great for most of my plan in a country that they don’t know it. We have to be so good and it’s a movement and we will make America strong again. Thank you. Thank you...
Project Objectives

- **Primary Objective**: Develop a character-level neural network that can generate text in the style of Donald Trump.
Project Objectives

➤ **Primary Objective**: Develop a character-level neural network that can generate text in the style of Donald Trump.

➤ **Secondary Objectives**:

1. Develop and compare novel *decision engine* algorithms for character selection.

2. Improve short-sequence generation through multi-length training.
Quick Review of Character-Level RNNs

- Given a sequence of characters, a character-level RNN learns a probability distribution over the possible subsequent characters.
Quick Review of Character-Level RNNs

- Given a sequence of characters, a character-level RNN learns a probability distribution over the possible subsequent characters.

- **Example:** if you give as input “We will build a great wal”, the RNN should return a distribution $p$ over a vocabulary of characters $V$ s.t. $p(‘l’)$ is large.
Quick Review of Character-Level RNNs

- Given a sequence of characters, a character-level RNN learns a probability distribution over the possible subsequent characters.

  **Example:** if you give as input “We will build a great wal”, the RNN should return a distribution $p$ over a vocabulary of characters $V$ s.t. $p(‘l’) \text{ is large}$

- Given some text, you can repeatedly choose a subsequent character based on the distribution produced by the network given the previous $L$ characters.
Embedding Matrix

\[
\begin{bmatrix}
W_{1,1} & \cdots & W_{1,|v|} \\
\vdots & \ddots & \vdots \\
W_{d,1} & \cdots & W_{d,|v|}
\end{bmatrix}
\]

One-Hot Vector Inputs

\[
\begin{bmatrix}
1 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & \ddots \\
0 & \ddots & \ddots & \ddots \\
0 & \cdots & \cdots & 0
\end{bmatrix}
\]
Learner Architecture – Summary

- Five Primary Stages:
  - One-Hot *Character* Encoding
  - Embedding Matrix
  - Multi-Layer LSTM
  - Feed-Forward Network
  - Softmax Layer
Learner Architecture – Summary

▶ Five Primary Stages:
  ▶ One-Hot Character Encoding
  ▶ Embedding Matrix
  ▶ Multi-Layer LSTM
  ▶ Feed-Forward Network
  ▶ Softmax Layer

▶ One-Hot & Softmax Dimension: ~95 Characters
Learner Architecture – Summary

- **Five Primary Stages:**
  - One-Hot *Character* Encoding
  - Embedding Matrix
  - Multi-Layer LSTM
  - Feed-Forward Network
  - Softmax Layer

- **One-Hot & Softmax Dimension:** ~95 Characters

- **LSTM:**
  - Two Layers
  - Hidden Layer Width: 128
  - Dropout – Surprisingly important! *(More details to come)*
Learner Architecture – Summary

- **Five Primary Stages:**
  - One-Hot *Character* Encoding
  - Embedding Matrix
  - Multi-Layer LSTM
  - Feed-Forward Network
  - Softmax Layer

- **One-Hot & Softmax Dimension:** ~95 Characters

- **LSTM:**
  - Two Layers
  - Hidden Layer Width: 128
  - Dropout – Surprisingly important! (*More details to come*)

- **Feed-Forward Network:**
  - One Hidden Layer with 256 Neurons
Discussion on Character-Level Text Generation

- **Question:** Wouldn’t a word-level RNN be better?

  - **Short Answer:** Word-level is not practical.

  - **Long Answer:**
    - Too many words (i.e., classes)
    - Limited hardware availability
    - Limited training time

  - **Question:** Is character-level text generation ideal?
  - **No.** We do not expect a character-level RNN to create perfectly coherent text.
  - It will only successfully mimic short phrases or at most a single paragraph.
Discussion on Character-Level Text Generation

▶ **Question:** Wouldn’t a word-level RNN be better?

▶ **Short Answer:** Word-level is not practical.
Question: Wouldn’t a word-level RNN be better?

Short Answer: Word-level is not practical.

Long Answer:
Discussion on Character-Level Text Generation

- **Question:** Wouldn’t a word-level RNN be better?
- **Short Answer:** Word-level is not practical.
- **Long Answer:**
  - Too many words (i.e., classes)
Discussion on Character-Level Text Generation

- **Question:** Wouldn’t a word-level RNN be better?

- **Short Answer:** Word-level is not practical.

- **Long Answer:**
  - Too many words (i.e., classes)
  - Limited hardware availability
Discussion on Character-Level Text Generation

- **Question:** *Wouldn’t a word-level RNN be better?*
- **Short Answer:** Word-level is not practical.
- **Long Answer:**
  - Too many words (i.e., classes)
  - Limited hardware availability
  - Limited training time
Discussion on Character-Level Text Generation

- **Question:** Wouldn’t a word-level RNN be better?
- **Short Answer:** Word-level is not practical.
- **Long Answer:**
  - Too many words (i.e., classes)
  - Limited hardware availability
  - Limited training time

- **Question:** Is character-level text generation ideal?
Discussion on Character-Level Text Generation

- **Question:** Wouldn’t a word-level RNN be better?
- **Short Answer:** Word-level is not practical.
- **Long Answer:**
  - Too many words (i.e., classes)
  - Limited hardware availability
  - Limited training time
- **Question:** Is character-level text generation ideal?
- **No.** We do not expect a character-level RNN to create perfectly coherent text.
  - It will only successfully mimic short phrases or at most a single paragraph.
Overview of the Training Dataset & Procedure

- **Datasets**
  - Approximately 115 speeches
  - Basic Statistics:
    - >365,000 Words
    - >2,000,000 Training Sequences

- **Speeches Only**

- **New Innovation: Variable length sequence training**
  - Rather than training only the maximum sequence length, we train intermediary sequence lengths to ensure quality outputs even on short sequences.
Text Generation Architecture

Embedding Matrix

\[
\begin{bmatrix}
 w_{1,1} & \cdots & w_{1,|v|} \\
 \vdots & \ddots & \vdots \\
 w_{d,1} & \cdots & w_{d,|v|}
\end{bmatrix}
\]

One-Hot Vector Inputs

\[
\begin{bmatrix}
 1 \\
 0 \\
 0 \\
 \vdots \\
 0
\end{bmatrix}
\begin{bmatrix}
 0 \\
 1 \\
 0 \\
 \vdots \\
 0
\end{bmatrix}
\begin{bmatrix}
 0 \\
 1 \\
 0 \\
 \vdots \\
 0
\end{bmatrix}
\]
Text Generation Architecture

Embedding Matrix

\[
\begin{bmatrix}
  w_{1,1} & \cdots & w_{1,|v|} \\
  \vdots & \ddots & \vdots \\
  w_{d,1} & \cdots & w_{d,|v|}
\end{bmatrix}
\]

One-Hot Vector Inputs

\[
\begin{bmatrix}
  1 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  0 \\
  1 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  0 \\
  1 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\]

Seed Text
Text Generation Architecture

Embedding Matrix

$$\begin{bmatrix}
  w_{1,1} & \cdots & w_{1,|v|} \\
  \vdots & \ddots & \vdots \\
  w_{d,1} & \cdots & w_{d,|v|}
\end{bmatrix}$$

One-Hot Vector Inputs

$$\begin{bmatrix}
  1 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix} \times \begin{bmatrix}
  0 \\
  1 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}$$

Seed Text
Text Generation Architecture

Embedding Matrix

\[
\begin{bmatrix}
w_{1,1} & \cdots & w_{1,|v|} \\
\vdots & \ddots & \vdots \\
w_{d,1} & \cdots & w_{d,|v|}
\end{bmatrix}
\]

One-Hot Vector Inputs

\[
\begin{bmatrix}
1 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
0 \\
1 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\]
Text Generation Architecture

Embedding Matrix

\[
\begin{bmatrix}
  w_{1,1} & \cdots & w_{1,|v|} \\
  \vdots & \ddots & \vdots \\
  w_{d,1} & \cdots & w_{d,|v|}
\end{bmatrix}
\]

One-Hot Vector Inputs

\[
\begin{bmatrix}
  1 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  0 \\
  1 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  0 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\]
Decision Engine

- **Greedy** *Always take the most probable character*
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - *The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want...*
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want...”

- **Random** *Select a random character according to the given distribution*
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “*The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to...*”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want...”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
  - Has a non-zero probability of doing something ridiculous
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want...”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
  - Has a non-zero probability of doing something ridiculous
  - “The media is so dishonest. And thank you. I trace it. We can change it. We don’t worts out.”
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want...”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
  - Has a non-zero probability of doing something ridiculous
  - “The media is so dishonest. And thank you. I trace it. We can change it. We don’t worts out.”

- **Top-k** *Make a random choice amongst the k most probable characters according to the sub-distribution*
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want...”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
  - Has a non-zero probability of doing something ridiculous
  - “The media is so dishonest. And thank you. I trace it. We can change it. We don’t worts out.”

- **Top-k** *Make a random choice amongst the k most probable characters according to the sub-distribution*
  - Throws out ridiculous random choices
Decision Engine

- **Greedy** *Always take the most probable character*
  - Always makes a confident choice
  - Leads to looping behavior
  - “The media is so dishonest. They want to stop the people of the world. I want to stop the people of the world. I want to stop the people of the world. I want…”

- **Random** *Select a random character according to the given distribution*
  - Avoids getting into loops
  - Has a non-zero probability of doing something ridiculous
  - “The media is so dishonest. And thank you. I trace it. We can change it. We don’t worts out.”

- **Top-k** *Make a random choice amongst the k most probable characters according to the sub-distribution*
  - Throws out ridiculous random choices
  - “The media is so dishonest. And they don’t know where you see it. I’m going to bring back the world.”
Random-Start + Greedy Finish *Make random choices for the first character of a word, then greedily finish each word*
Random-Start + Greedy Finish *Make random choices for the first character of a word, then greedily finish each word*

- Gets us out of infinite loops
Random-Start + Greedy Finish *Make random choices for the first character of a word, then greedily finish each word*

- Gets us out of infinite loops
- Does not mangle characters within words
Decision Engine Contd.

- **Random-Start + Greedy Finish** *Make random choices for the first character of a word, then greedily finish each word*
  - Gets us out of infinite loops
  - Does not mangle characters within words

- **Boosting Lopsidedness** *Exponentiate the distribution then re-normalize*
- **Random-Start + Greedy Finish** Make random choices for the first character of a word, then greedily finish each word
  - Gets us out of infinite loops
  - Does not mangle characters within words

- **Boosting Lopsidedness** Exponentiate the distribution then re-normalize
  - Boosts the chance of making a more probable choice
Decision Engine Contd.

- **Random-Start + Greedy Finish** *Make random choices for the first character of a word, then greedily finish each word*
  - Gets us out of infinite loops
  - Does not mangle characters within words

- **Boosting Lopsidedness** *Exponentiate the distribution then re-normalize*
  - Boosts the chance of a making a more probable choice

- **Sample with Dropout**
Decision Engine Contd.

- **Random-Start + Greedy Finish** *Make random choices for the first character of a word, then greedily finish each word*
  - Gets us out of infinite loops
  - Does not mangle characters within words

- **Boosting Lopsidedness** *Exponentiate the distribution then re-normalize*
  - Boosts the chance of making a more probable choice

- **Sample with Dropout**
  - Randomly change the network
Decision Engine Contd.

- **Random-Start + Greedy Finish** *Make random choices for the first character of a word, then greedily finish each word*
  - Gets us out of infinite loops
  - Does not mangle characters within words

- **Boosting Lopsidedness** *Exponentiate the distribution then re-normalize*
  - Boosts the chance of making a more probable choice

- **Sample with Dropout**
  - Randomly change the network

- **We can combine these with the algorithms above to create a wide variety of sampling methods**
Summary and Future Work

- **Summary**: We developed a character-level RNN that generated text in the style of President Donald Trump.

- **Future Work**:
  - **New Idea**: Character-Level Generation, Word-Level Decisions
Summary and Future Work

- **Summary**: We developed a character-level RNN that generated text in the style of President Donald Trump.

- **Future Work**:
  - **New Idea**: Character-Level Generation, Word-Level Decisions
  - **Training Improvements**: Given more time, we believe we could train better models.
    - Train longer sequences
    - Train entire speeches as a single unit
    - Train additional epochs (each additional epoch is )


Summary and Future Work
Summary and Future Work

- **Summary**: We developed a character-level RNN that generated text in the style of President Donald Trump.

- **Future Work**:
  - **New Idea**: Character-Level Generation, Word-Level Decisions
  - **Training Improvements**: Given more time, we believe we could train better models.
    - Train longer sequences
    - Train entire speeches as a single unit
    - Train additional epochs (each additional epoch is )
  - **Stretch Goal**: A chat bot so you can feel what it is like to have a conversation with Donald Trump.
Download the Source Code

Our full source code is available at:

https://github.com/ZaydH/trump_char_rnn