For dense input vectors (e.g., embeddings), full dot product

\[ h_1 = \sigma(x_1 W_1 + x_2 W_2 + x_3 W_3) \]
CNNs

- CNNs are limited to creating features scoped over a pre-defined width.

“Valentine’s Day is being marketed as a Date Movie. I think it’s more of a First-Date Movie. If your date likes it, do not date that person again. And if you like it, there may not be a second date.”

Roger Ebert, Valentine’s Day
If your date likes it, do not date that person again.
Recurrent neural network

- RNN allow arbitrarily-sized conditioning contexts; condition on the entire sequence history.
Recurrent neural network

- Often used for sequential prediction tasks:
  - Language models—predicting the next symbol (word, character) in a sequence
Generation

\[ \text{the} \xrightarrow{\text{predict}} y_1 \]
\[ \text{black} \xrightarrow{\text{predict}} y_2 \]
\[ \text{fox} \xrightarrow{\text{predict}} y_3 \]
\[ \text{jumped} \xrightarrow{\text{predict}} y_4 \]
\[ \langle/s\rangle \xrightarrow{\text{predict}} y_5 \]

\[ s_0 \xrightarrow{E[<s>]} R, O \]
\[ s_1 \xrightarrow{E[\text{the}]} R, O \]
\[ s_2 \xrightarrow{E[\text{black}]} R, O \]
\[ s_3 \xrightarrow{E[\text{fox}]} R, O \]
\[ s_4 \xrightarrow{E[\text{jumped}]} R, O \]
Character LM

/*
* Increment the size file of the new incorrect UI_FILTER group information
* of the size generatively.
*/

static int indicate_policy(void)
{
    int error;
    if (fd == MARN_EPT) {
       /*
        * The kernel blank will coeld it to userspace.
        */
       if (ss->segment < mem_total)
          unblock_graph_and_set_blocked();
       else
          ret = 1;
       goto bail;
    }

    segaddr = in_SB(in.addr);
    selector = seg / 16;
    setup_works = true;
    /*
     *
     */
}

http://karpathy.github.io/2015/05/21/rnn-effectiveness/
\begin{proof}
We may assume that $\mathcal{I}$ is an abelian sheaf on $\mathcal{C}$.
\begin{itemize}
  \item Given a morphism $\Delta : \mathcal{F} \to \mathcal{I}$
  is an injective and let $\mathfrak{q}$ be an abelian sheaf on $X$.
  Let $\mathcal{F}$ be a fibered complex. Let $\mathcal{F}$ be a category.
\end{itemize}
\begin{enumerate}
  \item \hyperref[setain-construction-phantom]{Lemma}
    \begin{enumerate}
    \item \hyperref[lemma-characterize-quasi-finite]{Lemma}
      Let $\mathcal{F}$ be an abelian quasi-coherent sheaf on $\mathcal{C}$.
      Let $\mathcal{F}$ be a coherent $\mathcal{O}_X$-module. Then
      $\mathcal{F}$ is an abelian catenary over $\mathcal{C}$.
    \item The following are equivalent
    \begin{enumerate}
    \item $\mathcal{F}$ is an $\mathcal{O}_X$-module.
    \end{enumerate}
    \end{enumerate}
  \end{enumerate}
\end{proof}

http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Character LM

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Recurrent neural network

• Often used for sequential prediction tasks:
  
  • Language models—predicting the next symbol (word, character) in a sequence
  
  • Machine translation—predicting a sequence of words (sentence) in language $f$ conditioned on sentence in language $e$
Lasciate ogni speranza, voi ch'entrate

Abandon all hope, ye who enter here
Je suis heureux

I'm EOS

I'm happy
Recurrent neural network

- Often used for sequential prediction tasks:
  - Language models—predicting the next symbol (word, character) in a sequence
  - Machine translation—predicting a sequence of words (sentence) in language $f$ conditioned on sentence in language $e$
  - Sequence labeling (POS tagging, NER)—we’ll cover that after spring break.
For POS tagging, predict the tag from \( y \) conditioned on the context.
We’ll use them today to generate a representation of a sequence of arbitrary length (sentence, document, etc.) optimized for some task.

We can then use that representation for e.g. text classification/regression.
Recurrent neural network
Each y is the output of the RNN at that time step; sometimes we use this information (POS tagging, LM); sometimes we only use the output for the final state (s₅)

Each x here is one token in a sequence
Each time step has two inputs:

- $x_i$ (the observation at time step $i$); one-hot vector, feature vector or word embedding.
- $s_{i-1}$ (the output of the previous state); base case: $s_0 = 0$ vector
Recurrent neural network

\[ s_i = R(x_i, s_{i-1}) \]

\[ y_i = O(s_i) \]

R computes the output state as a function of the current input and previous state.

O computes the output as a function of the current output state.
“Simple” RNN

\[ s_i = R(x_i, s_{i-1}) = g(s_{i-1}W^s + x_iW^x + b) \]

Different weight vectors \( W \) transform the previous state and current input before combining:

- \( W^s \in \mathbb{R}^{H \times H} \)
- \( W^x \in \mathbb{R}^{D \times H} \)
- \( b \in \mathbb{R}^H \)

\[ y_i = O(s_i) = s_i \]

Elman 1990, Mikolov 2012
How do we use RNNs for document classification?
I loved the movie!
RNNs

• The final hidden state contains a representation informed by the entire sequence, including non-linear interactions between the elements of that sequence.
If your date likes it, do not date that person again in conditional until comma, so downplay “likes”

The director said it was “the best movie he ever made” quoted speech is in between quotation marks, so downplay “best”
RNNs

- Recurrent networks are deep in that they involve one “layer” for each time step (e.g., words in a sentence)

- **Vanishing gradient problem**: as error is back propagated through the layers of a deep network, they tend toward 0.
Long short-term memory network (LSTM)

• Designed to account for the vanishing gradient problem

• Basic idea: split the $s$ vector propagated between time steps into a memory component and a hidden state component
LSTMs

http://colah.github.io/posts/2015-08-Understanding-LSTMs/
Gates

- LSTMs gates control the flow of information

- A sigmoid squashes its input to between 0 and 1
- By multiplying the output of a sigmoid elementwise with another vector, we forget information in the vector (if multiplied by a number close to 0) or allow it to pass (if multiplied by number close to 1)
Forget gate: as a function of the previous hidden state and current input, forget information in the memory
Input gate (but forget some information about the current observation)

| 0.03 | 1.4 | -0.35 | -1.38 | 0.08 |

memory

hidden state

elementwise product

$h_t$
Update the memory (but forget some information about the current observation)
The memory passes directly to the next state

| 0.8 | 0.4 | -1.4 | 9.8 | 3.4 |
Output gate: forget some information to send to the hidden state
The hidden state is updated with the current observation and new context.
How much context?

- For language modeling, LSTMs are aware of about 200 words of context
- Ignores word order beyond 50 words
Context

- Encoding an entire sequence into a fixed dimensional vector at the end of the sequence can potentially lose a lot of information, especially for documents.

- Rarely used in practice.
I loved the movie!
I loved the movie!
I loved the movie!

Iyyer et al. (2015), “Deep Unordered Composition Rivals Syntactic Methods for Text Classification” (ACL)
I loved the movie!
RNN

• With an RNN, we can generate a representation of the sequence as seen through time $t$.

• This encodes a representation of meaning specific to the local context a word is used in.
We can then swap that RNN time step output for the embeddings we used earlier.
What about the future context?
Bidirectional RNN

• A powerful alternative is make predictions conditioning both on the past and the future.

• Two RNNs
  • One running left-to-right
  • One right-to-left

• Each produces an output vector at each time step, which we concatenate
Bidirectional RNN

forward RNN
Bidirectional RNN

I loved the movie!
Bidirectional RNN
Bidirectional RNN

- The forward RNN and backward RNN each output a vector of size $H$ at each time step, which we concatenate into a vector of size $2H$.

- The forward and backward RNN each have separate parameters to be learned during training.
Training BiRNNs

• Given this definition of an BiRNN:

\[
\begin{align*}
    s_b^i &= R_b(x^i, s_b^{i+1}) = g(s_b^{i+1}W_b^s + x^iW_b^x + b_b) \\
    s_f^i &= R_f(x^i, s_f^{i-1}) = g(s_f^{i-1}W_f^s + x^iW_f^x + b_f) \\
    y_i &= \text{softmax} \left([s_f^i; s_b^i]W^o + b^o\right)
\end{align*}
\]

• We have 8 sets of parameters to learn (3 for each RNN + 2 for the final layer)
Many neural network libraries require each sequence within the same batch to be the same length.

We can make artificially make this so by padding shorter sequences with a special symbol not otherwise used (e.g. 0)
<table>
<thead>
<tr>
<th>the</th>
<th>dog</th>
<th>ran</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>he</td>
<td>ran</td>
<td>to</td>
<td>the</td>
<td>house</td>
</tr>
<tr>
<td>he</td>
<td>stopped</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>he</td>
<td>went</td>
<td>inside</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

word embedding ids
<table>
<thead>
<tr>
<th>1</th>
<th>3</th>
<th>4</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

word embedding ids
I loved it.

Padding can cause problems unless you account for it.
Masking

- For sequences that have been padded, **ignore** all time steps with the padding symbol.
Activity

• Explore LSTMs with 8.neural/LSTM.ipynb