Sequence to Sequence models

Rasmus Berg Palm

Technical University of Denmark & Tradeshift
Sequence to Sequence models

\[ x \rightarrow \text{Model} \rightarrow y \]
Sequence to Sequence models

\[ x: \text{“the dog ate my homework”} \]

\[ y: \text{“El perro se comió mi tarea”} \]
Sequence to Sequence models

$$\mathbf{x}: [x_1, x_2, \ldots, x_n]$$

$$x_i: \text{one-hot encoded word}$$

$$\mathbf{x}.\text{shape} = [n, \text{x-vocab}]$$
Sequence to Sequence models

\[ y: [y_1, y_2, \ldots, y_m] \]

\[ y_i: \text{one-hot encoded word} \]

\[ y.\text{shape} = [m, y\text{-vocab}] \]
Sequence to Sequence models


Sequence to Sequence models

1. Squeeze the entire \( \mathbf{x} \) into a single vector \( \mathbf{v} \)

2. Generate \( \mathbf{y} \) conditioned on \( \mathbf{v} \)
1. Squeeze the entire $\mathbf{x}$ into a single vector $\mathbf{v}$
Sequence to Sequence models

Bag of Words

\[ v = \text{sum}(x) \]

nah...
Sequence to Sequence models

Bag of Embeddings

\[ v = \text{sum}(\text{embed}(x)) \]

nah...
Sequence to Sequence models

LSTM encoding

\[ v = \text{LSTM}(x)[-1] \]

yup!
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]

Deep LSTM

Input sequence
Sequence to Sequence models

1. Squeeze the entire $x$ into a single vector $v$

2. Generate $y$ conditioned on $v$

Ideas?
Sequence to Sequence models

\[ y = \text{LSTM}^* \]

*Set initial hidden state to \( v \)
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]
Sequence to Sequence models

Variant that is easier to code (and better)
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]
Sequence to Sequence models

OK.

Let’s do it!

https://github.com/rasmusbergpalm/normalization
Sequence to Sequence models

\( x: 12 \text{ November 2016} \)

\( y: 2016-11-12 \)
Sequence to Sequence models

```python
# Encoder
source = Input(shape=(None,), dtype='int32', name='source')
embedded = Embedding(output_dim=128, input_dim=train.source_vocab_size(), mask_zero=True)(source)
last_hid = LSTM(output_dim=128)(embedded)

# Decoder
repeated = RepeatVector(train.target.padded.shape[1])(last_hid)
decoder = LSTM(output_dim=128, return_sequences=True)(repeated)
output = TimeDistributed(Dense(output_dim=train.target_vocab_size(), activation='softmax'))(decoder)
model = Model([source], output=[output])
```
Sequence to Sequence models

http://localhost:5000
Sequence to Sequence models

Trick

Feed the LSTM the last output it made
Sequence to Sequence models

Two ways to implement

1. Feed the actual probabilities outputted
   Hard, not used very often
2. Feed the target shifted by one
   Easy, very used. AKA. “teacher forcing”
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]

Deep LSTM

Input sequence

Target sequence
Sequence to Sequence models

But...

How to generate at test time then?
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]

Deep LSTM

Input sequence

Target sequence
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]
Sequence to Sequence models

[Sutskever & Vinyals & Le NIPS 2014]

Deep LSTM

Input sequence

Target sequence

A → B → C → D → v

X → Y → Z → Q

v → v → v → v → v
Sequence to Sequence models

Exercise left for the reader

Implement teacher forcing in the date parser
Sequence to Sequence models

The great weakness.

Ideas?
Sequence to Sequence models

The great weakness.

\[ \mathbf{x} \text{ is } n \text{ long} \]

\[ \mathbf{v} \text{ is fixed size} \]

Hard to compress when \( n \) grows
Sequence to Sequence models

![Graph showing BLEU score vs sentence length for Source text, Reference text, and Both, indicating performance trends.](image-url)
Sequence to Sequence models

The great solution.

Ideas?
Sequence to Sequence models


https://arxiv.org/abs/1409.0473
Sequence to Sequence models

Let the decoder look at the entire input sequence for every output

AKA. “Attention”
Sequence to Sequence models
Sequence to Sequence models

Attention is tricky…

But you’re clever :]

But you’re clever :]
\[ a_i = f(e_i, d_j) \]
$w_i = \text{softmax}(a_i) \ (1,)$

$a_i = f(e_i, d_j) \ (1,)$
w_i = softmax(a_i) (1,)

a_i = f(e_i, d_j) (1,)
\[ c_i = w_i e_i \ (128,) \]
\[ w_i = \text{softmax}(a_i) \ (1,) \]
\[ a_i = f(e_i, d_j) \ (1,) \]
\[ C_1 = \text{sum}(c_i) \quad (128,) \]

\[ c_i = w_i e_i \quad (128,) \]

\[ w_i = \text{softmax}(a_i) \quad (1,) \]

\[ a_i = f(e_i, d_j) \quad (1,) \]
\[ c_1 = \text{sum}(c_i) \ (128,) \]
\[ c_i = w_i e_i \ (128,) \]
\[ w_i = \text{softmax}(a_i) \ (1,) \]
\[ a_i = f(e_i, d_j) \ (1,) \]
\[ c_1 = \text{sum}(c_i) \ (128,) \]
\[ c_i = w_i e_i \ (128,) \]
\[ w_i = \text{softmax}(a_i) \ (1,) \]
\[ a_i = f(e_i, d_j) \ (1,) \]
\[ c_2 = \text{sum}(c_i) \ (128,) \]
\[ c_i = w_i e_i \ (128,) \]
\[ w_i = \text{softmax}(a_i) \ (1,) \]
\[ a_i = f(e_i, d_j) \ (1,) \]
\[ C_2 = \text{sum}(c_i) \quad (128,) \]
\[ c_i = w_i e_i \quad (128,) \]
\[ w_i = \text{softmax}(a_i) \quad (1,) \]
\[ a_i = f(e_i, d_j) \quad (1,) \]
\[ c_2 = \text{sum}(c_i) \ (128,) \]

\[ c_i = w_i e_i \ (128,) \]

\[ w_i = \text{softmax}(a_i) \ (1,) \]

\[ a_i = f(e_i, d_j) \ (1,) \]
Sequence to Sequence models

Phew!
Sequence to Sequence models
Sequence to Sequence models

![Graph showing BLEU score vs. sentence length for different models](image)
Sequence to Sequence models

Exercise left for the reader
Implement attention for the date parser
Sequence to Sequence models

Trick

Teacher forcing makes attention easier to implement
My own work
The actual data and problem

<table>
<thead>
<tr>
<th>ITEM REF.</th>
<th>ITEM DESCRIPTION</th>
<th>QTY</th>
<th>PRICE</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF673</td>
<td>Flux capacitor (DeLorean compatible)</td>
<td>1</td>
<td>95.00</td>
<td>95.00</td>
</tr>
<tr>
<td>AS245</td>
<td>Cell phone Samsung SPH-N770</td>
<td>10</td>
<td>345.00</td>
<td>3450.00</td>
</tr>
<tr>
<td>ZS304</td>
<td>Psychromagnetic Slime</td>
<td>100</td>
<td>17.00</td>
<td>1700.00</td>
</tr>
</tbody>
</table>

**SUBTOTAL:** USD 100.150.00  
**TAX (20% VAT):** USD 20.030.00  
**TOTAL:** USD 120.170.00

The payment must be done 14 days after the invoice date, while any claim must be done within 10 days. Burn of other words in very small text that not easy read. They do not follow any formal and may repeat text that it was previously included in the invoice. For our purpose, there is nothing interesting here.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Invoice>
    <Number>123456</Number>
    <Date>2015-02-14</Date>
    <Sender>Tyrell Corporation</Sender>
    <Receiver>Stark Industries</Receiver>
    <TaxableAmount currency="USD">100150.00</TaxableAmount>
    <TaxPercent>20.00</TaxPercent>
    <TaxAmount currency="USD">20030.00</TaxAmount>
    <Total currency="USD">120170.00</Total>
</Invoice>
```
Interesting challenges

1. Training data is PDF and XML pairs. No word annotations!
2. Large handcrafted post-processing stage
3. Structured output (totals have to add up, etc.)
4. Modelling word context
5. Using image features
**TYRELL CORPORATION**
Dr. Eldon Street, 1, Los Angeles, CA 91020, USA

**STARK INDUSTRIES**
Howard Stark, 40
Palmdale,
CA 93550, USA
98-7654321

---

<table>
<thead>
<tr>
<th>ITEM REF.</th>
<th>ITEM DESCRIPTION</th>
<th>QTY</th>
<th>PRICE</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF673</td>
<td>Flux capacitor (DeLorean compatible)</td>
<td>1</td>
<td>95,000,-</td>
<td>95,000,-</td>
</tr>
<tr>
<td>AS245</td>
<td>Cell phone Samsung SPH-N70</td>
<td>10</td>
<td>345,-</td>
<td>3450,-</td>
</tr>
<tr>
<td>ZS304</td>
<td>Psychromagnetonic Slime</td>
<td>100 l</td>
<td>17,00</td>
<td>1700,-</td>
</tr>
</tbody>
</table>

---

**SUBTOTAL:** USD 100,150.00
**TAX (20% VAT):** USD 20,030.00

**TOTAL:** USD 120,180.00

The payment must be done 14 days after the invoice date, while any claim must be done within 10 days. bunch of other words in very small font that make the reader. They do not follow any format and may repeat text that the previously included in the invoice. For our purpose, there is nothing interesting here.

---

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Invoice>
  <Number>123456</Number>
  <Date>2015-02-14</Date>
  <Sender>Tyrell Corporation</Sender>
  <Receiver>Stark Industries</Receiver>
  <TaxableAmount currency="USD">100150.00</TaxableAmount>
  <TaxPercent>20.00</TaxPercent>
  <TaxAmount currency="USD">20030.00</TaxAmount>
  <Total currency="USD">120170.00</Total>
</Invoice>
```
Addressing the lack of word annotations

‘End-to-End Information Extraction without Token-Level Supervision’

Rasmus Berg Palm, Dirk Hovy, Ole Winther, Florian Laws

https://arxiv.org/abs/1707.04913
Let’s create a travel concierge app

Takes natural language input.

Proposes flights.
Behind the covers..

We use a flight search engine.

The search engine accepts a fixed set of fields, e.g. “from”, “to”, “day”, etc.

We return the top-1 hit.
Behind the covers...

We hire human operators to extract the values for the search engine fields. This labor is tedious for the operators and costly for us.
So let’s automate this extraction of information
<table>
<thead>
<tr>
<th>Token</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>cheapest</td>
<td>B-PRICE</td>
</tr>
<tr>
<td>airfare</td>
<td>O</td>
</tr>
<tr>
<td>from</td>
<td>O</td>
</tr>
<tr>
<td>tacoma</td>
<td>B-FROM</td>
</tr>
<tr>
<td>to</td>
<td>O</td>
</tr>
<tr>
<td>st.</td>
<td>B-TO</td>
</tr>
<tr>
<td>louis</td>
<td>I-TO</td>
</tr>
<tr>
<td>Token</td>
<td>Label</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>cheapest</td>
<td>B-PRICE</td>
</tr>
<tr>
<td>airfare</td>
<td>O</td>
</tr>
<tr>
<td>from</td>
<td>O</td>
</tr>
<tr>
<td>tacoma</td>
<td>B-FROM</td>
</tr>
<tr>
<td>to</td>
<td>O</td>
</tr>
<tr>
<td>st.</td>
<td>B-TO</td>
</tr>
<tr>
<td>louis</td>
<td>I-TO</td>
</tr>
<tr>
<td>Token</td>
<td>Label</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>cheapest</td>
<td>B-PRICE</td>
</tr>
<tr>
<td>airfare</td>
<td>O</td>
</tr>
<tr>
<td>from</td>
<td>O</td>
</tr>
<tr>
<td>tacoma</td>
<td>B-FROM</td>
</tr>
<tr>
<td>to</td>
<td>O</td>
</tr>
<tr>
<td>st.</td>
<td>B-TO</td>
</tr>
<tr>
<td>louis</td>
<td>I-TO</td>
</tr>
<tr>
<td>Token</td>
<td>Label</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
</tr>
<tr>
<td>cheapest</td>
<td>B-PRICE</td>
</tr>
<tr>
<td>airfare</td>
<td>O</td>
</tr>
<tr>
<td>from</td>
<td>O</td>
</tr>
<tr>
<td>tacoma</td>
<td>B-FROM</td>
</tr>
<tr>
<td>to</td>
<td>O</td>
</tr>
<tr>
<td>st.</td>
<td>B-TO</td>
</tr>
<tr>
<td>louis</td>
<td>I-TO</td>
</tr>
</tbody>
</table>
Our model

Encoder

Decoder

Decoder

cheapest  airfare  from  tacoma  to  st.  louis  <EOS>

<GO>  st.  louis

<GO>  tacoma
Our model
Our model
Our model
Our model
Our model
Our model

Encoder

Decoder$_1$

Decoder$_2$
Our model
Our model
Our model
Code is available at

github.com/rasmusbergpalm/e2e-ie-release
There’s one major limitation

Normalization

‘17 Jan 2012’ → ‘2012-01-17’