Learning to Search
Reminders

• Homework 1: Neural Networks for Sequence Tagging
  – Out: Wed, Sep 7 (later today!)
  – Due: Fri, Sep 16 at 11:59pm
  – Two parts:
    1. written part to Gradescope (Written slot)
    2. programming part to Gradescope (Programming slot)
EXAMPLE SEQ2SEQ ARCHITECTURES
Example Architectures

**deep LSTM + RNN-LM**

- **Encoder**: three-layer unidirectional LSTM
- **Decoder**: a one-layer RNN-LM

```
Vamos  al  cafe  ahora

b_1    b_2    b_3    b_4  

c_1    c_2    c_3    c_4

b_1    b_2    b_3    b_4

c_1    c_2    c_3    c_4

RNN-LM

d_1    d_2    d_3

p(w_3|h_3)

to
```

Let's go
Example Architectures

depth LSTM + deep RNN-LM
• *Encoder*: three-layer unidirectional LSTM
• *Decoder*: a two-layer RNN-LM
Example Architectures

**biLSTM + deep RNN-LM**

- **Encoder**: two-layer bidirectional LSTM
- **Decoder**: a two-layer RNN-LM
LEARNING A SEQ2SEQ MODEL
Comparing RNN, RNN-LM, seq2seq

Whiteboard:

– Objective functions for RNN, RNN-LM, and seq2seq models
– Training a seq2seq model
DECODING FOR SEQ2SEQ MODELS
Decoding for seq2seq Models

At test time, how do we obtain predictions from our model?

• The two most common approaches:
  – Greedy search
  – Beam search

• Many alternatives:
  – Ancestral sampling (assuming we have a locally normalized model)
  – Nucleus sampling
  – Top-k sampling
Background: Greedy Search

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf

Greedy Search:
- At each node, selects the edge with lowest (immediate) weight
- **Heuristic** method of search (i.e. does not necessarily find the best path)
- Computation time: **linear** in max path length
Background: Greedy Search

Greedy Search:
- At each node, selects the edge with lowest (immediate) weight
- Heuristic method of search (i.e. does not necessarily find the best path)
- Computation time: linear in max path length

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf
Background: Greedy Search

Greedy Search:
- At each node, selects the edge with lowest (immediate) weight
- **Heuristic** method of search (i.e. does not necessarily find the best path)
- Computation time: linear in max path length

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf
Background: Beam Search

Beam Search:
- The “beam” is current set of best k nodes
- Let the expansion set be all neighbors of nodes in the beam
- At each time step, selects the set of k nodes in the expansion set with lowest (immediate) weight
- **Heuristic** method of search (i.e. does not necessarily find the best path)
- Computation time: **linear** in max path length

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf

Beam Size = 2

Goal:
- Search space consists of nodes and weighted edges
- Goal is to find the lowest (total) weight path from root to a leaf
Decoding for seq2seq Models

At test time, how do we obtain predictions from our model?

• The two most common approaches:
  – Greedy search
  – Beam search

• Many alternatives:
  – Ancestral sampling (assuming we have a locally normalized model)
  – Nucleus sampling
  – Top-k sampling

**Important Observation**

• maximum likelihood training (MLE) assumes that our inference strategy will return the **highest probability sequence**

• at **test time**, our **inference** strategies are all **heuristic** (i.e. they will make mistakes)
APPLICATIONS OF SEQ2SEQ
seq2seq for MT

**Basic Architecture:**

Figure 1: **Neural machine translation** – a stacking recurrent architecture for translating a source sequence A B C D into a target sequence X Y Z. Here, `<eos>` marks the end of a sentence.

*Results from Sutskever et al. (2014)*

<table>
<thead>
<tr>
<th>Method</th>
<th>test BLEU score (ntst14)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bahdanau et al. [2]</td>
<td>28.45</td>
</tr>
<tr>
<td>Baseline System [29]</td>
<td>33.30</td>
</tr>
<tr>
<td>Single forward LSTM, beam size 12</td>
<td>26.17</td>
</tr>
<tr>
<td>Single reversed LSTM, beam size 12</td>
<td>30.59</td>
</tr>
<tr>
<td>Ensemble of 5 reversed LSTMs, beam size 1</td>
<td>33.00</td>
</tr>
<tr>
<td>Ensemble of 2 reversed LSTMs, beam size 12</td>
<td>33.27</td>
</tr>
<tr>
<td>Ensemble of 5 reversed LSTMs, beam size 2</td>
<td>34.50</td>
</tr>
<tr>
<td>Ensemble of 5 reversed LSTMs, beam size 12</td>
<td><strong>34.81</strong></td>
</tr>
</tbody>
</table>

Table: performance on WMT’14 English to French test set

*Visualization from Sutskever et al. (2014)*

Figure 2: The figure shows a 2-dimensional PCA projection of the LSTM hidden states that are obtained after processing the phrases in the figures. The phrases are clustered by meaning, which in these examples is primarily a function of word order, which would be difficult to capture with a bag-of-words model. Notice that both clusters have similar internal structure.
seq2seq for ASR

Figure 1: LAS model.

\[ h = \text{Listen}(x) \]
\[ P(y_i|x, y_{<i}) = \text{AttendAndSpell}(y_{<i}, h) \]
# seq2seq for ASR

**Listen Attend and Spell**

**Results from Park et al. (2019)**

<table>
<thead>
<tr>
<th>Method</th>
<th>No LM</th>
<th>With LM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>clean</td>
<td>other</td>
</tr>
<tr>
<td><strong>HMM</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Povey et al., (2016) [29]</td>
<td>4.28</td>
<td></td>
</tr>
<tr>
<td>Han et al., (2017) [30]</td>
<td>3.51</td>
<td>8.58</td>
</tr>
<tr>
<td>Yang et al. (2018) [31]</td>
<td>2.97</td>
<td>7.50</td>
</tr>
<tr>
<td><strong>CTC/ASG</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collobert et al., (2016) [32]</td>
<td>7.2</td>
<td></td>
</tr>
<tr>
<td>Liptchinsky et al., (2017) [33]</td>
<td>6.7</td>
<td>20.8</td>
</tr>
<tr>
<td>Zhou et al., (2018) [34]</td>
<td>5.42</td>
<td></td>
</tr>
<tr>
<td>Li et al., (2019) [36]</td>
<td>3.86</td>
<td>11.95</td>
</tr>
<tr>
<td><strong>LAS</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zeyer et al., (2018) [23]</td>
<td>4.87</td>
<td>15.39</td>
</tr>
<tr>
<td>Zeyer et al., (2018) [37]</td>
<td>4.70</td>
<td>15.20</td>
</tr>
<tr>
<td>Sabour et al., (2019) [38]</td>
<td>4.5</td>
<td>13.3</td>
</tr>
</tbody>
</table>

---

**Fig. 1:** Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence \( x \) into high level features \( h \), the speller is an attention-based decoder generating the \( y \) characters from \( h \).
seq2seq for ASR

### Listen Attend and Spell

![Figure 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence \( x \) into high level features \( h \), the speller is an attention-based decoder generating the \( y \) characters from \( h \).](image)

### Results from Park et al. (2019)

Table 3: *LibriSpeech 960h WERs (%)*.  

<table>
<thead>
<tr>
<th>Method</th>
<th>No LM</th>
<th>With LM</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>clean</td>
<td>other</td>
<td>clean</td>
<td>other</td>
<td>clean</td>
<td>other</td>
<td>clean</td>
<td>other</td>
</tr>
<tr>
<td><strong>HMM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Povey et al., (2016) [29]</td>
<td>4.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Han et al., (2017) [30]</td>
<td>3.51</td>
<td>8.58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yang et al., (2018) [31]</td>
<td>2.97</td>
<td>7.50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CTC/ASG</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collobert et al., (2016) [32]</td>
<td>7.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Liptchinsky et al., (2017) [33]</td>
<td>6.7</td>
<td>20.8</td>
<td>4.8</td>
<td>14.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zhou et al., (2018) [34]</td>
<td>5.42</td>
<td>14.70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Li et al., (2019) [36]</td>
<td>3.86</td>
<td>11.95</td>
<td>2.95</td>
<td>8.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LAS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zeyer et al., (2018) [23]</td>
<td>4.87</td>
<td>15.39</td>
<td>3.82</td>
<td>12.76</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zeyer et al., (2018) [37]</td>
<td>4.70</td>
<td>15.20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sabour et al., (2019) [38]</td>
<td>4.5</td>
<td>13.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Our Work</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAS</td>
<td>4.1</td>
<td>12.5</td>
<td>3.2</td>
<td>9.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAS + SpecAugment</td>
<td>2.8</td>
<td>6.8</td>
<td>2.5</td>
<td>5.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Park et al. (2019) used the **LAS model** from prior work, and introduced a **data augmentation** method that gave state-of-the-art performance on LibriSpeech 960h and Switchboard 300h tasks.
Image Captioning

\[ p(\text{English} \mid \text{French}) \]

\[ \downarrow \]

\[ p(\text{English} \mid \text{Image}) \]


Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Image Captioning

\[ \theta^* = \arg \max_{\theta} p(S|I) \]

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Image Captioning

Human: A close up of two bananas with bottles in the background.

BestModel: A bunch of bananas and a bottle of wine.

InitialModel: A close up of a plate of food on a table.

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Image Captioning

Human: A woman holding up a yellow banana to her face.

BestModel: A woman holding a banana up to her face.

InitialModel: A close up of a person eating a hot dog.

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Human: A man outside cooking with a sub in his hand.

BestModel: A man is holding a sandwich in his hand.

InitialModel: A man cutting a cake with a knife.

Image Captioning

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Image Captioning

Human: Someone is using a small grill to melt his sandwich.

BestModel: A person is cooking some food on a grill.

InitialModel: A pizza sitting on top of a white plate.

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Image Captioning

Human: A blue, yellow and red train travels across the tracks near a depot.

BestModel: A blue and yellow train traveling down train tracks.

InitialModel: A train that is sitting on the tracks.

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Learning Objectives

Sequence to Sequence Models

You should be able to...

1. Apply an RNN to time-series structured prediction tasks
2. Employ an RNN-LM for various structured prediction tasks through prompting
3. Explain the difference between RNNs, RNNLMs, encoder-decoder models, and seq2seq models
4. Implement and train a basic seq2seq model
IMITATION LEARNING
# Imitation Learning vs. RL

<table>
<thead>
<tr>
<th>Paradigm</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td>$D = {x^{(i)}, y^{(i)}}_{i=1}^{N}$ \quad x \sim p^<em>(\cdot) and y = c^</em>(\cdot)$</td>
</tr>
<tr>
<td>Regression</td>
<td>$y^{(i)} \in \mathbb{R}$</td>
</tr>
<tr>
<td>Classification</td>
<td>$y^{(i)} \in {1, \ldots, K}$</td>
</tr>
<tr>
<td>Binary classification</td>
<td>$y^{(i)} \in {+1, -1}$</td>
</tr>
<tr>
<td>Structured Prediction</td>
<td>$y^{(i)}$ is a vector</td>
</tr>
<tr>
<td>Unsupervised</td>
<td>$D = {x^{(i)}}_{i=1}^{N}$</td>
</tr>
<tr>
<td>Semi-supervised</td>
<td>$D = {x^{(i)}, y^{(i)}}<em>{i=1}^{N_1} \cup {x^{(j)}}</em>{j=1}^{N_2}$</td>
</tr>
<tr>
<td>Online</td>
<td>$D = {(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots}$</td>
</tr>
<tr>
<td>Active Learning</td>
<td>$D = {x^{(i)}}_{i=1}^{N}$ and can query $y^{(i)} = c^*(\cdot)$ at a cost</td>
</tr>
<tr>
<td>Imitation Learning</td>
<td>$D = {(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots}$</td>
</tr>
<tr>
<td>Reinforcement Learning</td>
<td>$D = {(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \ldots}$</td>
</tr>
</tbody>
</table>
Autonomous Driving via Imitation Learning

- **Goal**: learn to **drive a car** around a **dirt track** at **high speed** without crashing
- **Approach 1**: (Williams et al., 2016; 2017)
  - model-predictive control (MPC)
  - expensive, accurate sensors required:
    - Global Positioning System (GPS)
    - Inertial Measurement Unit (IMU)
  - effective, but limited applicability
- **Approach 2**: (Pan et al., 2018)
  - imitation learning with deep CNN defining the policy
  - low-cost, on-board sensors:
    - monocular camera
    - wheel speed sensors
  - learn from expert demonstrations to reduce risk of crash
Autonomous Driving via Imitation Learning
Autonomous Driving via Imitation Learning

Why is this hard?
Imitation Learning

state (sensors)

agent (car)

policy (neural network)

action (left / right)

Figures from Pan et al. (2018)
Imitation Learning

Whiteboard:

– Fully supervised imitation learning
– The pitfall of fully supervised imitation learning
– DAgger for imitation learning