Sequence to Sequence Models
Q: What did the results of the survey look like?

A: Responses are still coming in, but one trend is clearly emerging: 75% of you already know HMMs.
Q: What is the difference between imitation learning and reinforcement learning?

A: There are lots of differences but they all stem from one fundamental difference:

Imitation learning assumes that it has access to an oracle policy $\pi^*$, reinforcement learning does not.

Interesting contrast: Q-Learning vs. DAgger.

- both have some notion of explore/exploit (very loose analogy)
- but Q-learning’s exploration is random, and its exploitation relies on the model’s policy
- whereas DAgger exploration uses the model’s policy, and its exploitation follows the oracle
Reminders

• Homework 1: DAgger for seq2seq
  – Out: Wed, Sep. 11 (+/- 2 days)
  – Due: Wed, Sep. 25 at 11:59pm
SEQ2SEQ: OVERVIEW
Why seq2seq?

- **10 years ago:** state-of-the-art machine translation or speech recognition systems were complex pipelines
  - **MT**
    - unsupervised word-level alignment of sentence-parallel corpora (e.g. via GIZA++)
    - build phrase tables based on (noisily) aligned data (use prefix trees and on demand loading to reduce memory demands)
    - use factored representation of each token (word, POS tag, lemma, morphology)
    - learn a separate language model (e.g. SRILM) for target
    - combine language model with phrase-based decoder
    - tuning via minimum error rate training (MERT)
  - **ASR**
    - MFCC and PLP feature extraction
    - acoustic model based on Gaussian Mixture Models (GMMs)
    - model phones via Hidden Markov Models (HMMs)
    - learn a separate n-gram language model
    - learn a phonetic model (i.e. mapping words to phones)
    - combine language model, acoustic model, and phonetic model in a weighted finite-state transducer (WFST) framework (e.g. OpenFST)
    - decode from a confusion network (lattice)
- **Today:** just use a seq2seq model
  - **encoder:** reads the input one token at a time to build up its vector representation
  - **decoder:** starts with encoder vector as context, then decodes one token at a time – feeding its own outputs back in to maintain a vector representation of what was produced so far
Outline

• Recurrent Neural Networks
  – Elman network
  – Backpropagation through time (BPTT)
  – Parameter tying
  – bidirectional RNN
  – Vanishing gradients
  – LSTM cell
  – Deep RNNs
  – Training tricks: mini-batching with masking, sorting into buckets of similar-length sequences, truncated BPTT

• Sequence-to-sequence (seq2seq) models
  – encoder-decoder architectures
  – Example: biLSTM + RNNLM
  – Learning to Search for seq2seq
    • DAgger for seq2seq
    • Scheduled Sampling (a special case of DAgger)
  – Example: machine translation
  – Example: speech recognition
  – Example: image captioning

• RNN Language Models
  – Definition: language modeling
  – n-gram language model
  – RNNLM
RECURRENT NEURAL NETWORKS
Long Short-Term Memory (LSTM)

Motivation:
• Standard RNNs have trouble learning long distance dependencies
• LSTMs combat this issue
Long Short-Term Memory (LSTM)

Motivation:
• Vanishing gradient problem for Standard RNNs
• Figure shows sensitivity (darker = more sensitive) to the input at time $t=1$

Figure from (Graves, 2012)
Long Short-Term Memory (LSTM)

Motivation:
- LSTM units have a rich internal structure
- The various “gates” determine the propagation of information and can choose to “remember” or “forget” information

Figure from (Graves, 2012)
Long Short-Term Memory (LSTM)
Long Short-Term Memory (LSTM)

- **Input gate**: masks out the standard RNN inputs
- **Forget gate**: masks out the previous cell
- **Cell**: stores the input/forget mixture
- **Output gate**: masks out the values of the next hidden

\[
i_t = \sigma \left( W_{xi} x_t + W_{hi} h_{t-1} + W_{ci} c_{t-1} + b_i \right)
\]
\[
f_t = \sigma \left( W_{xf} x_t + W_{hf} h_{t-1} + W_{cf} c_{t-1} + b_f \right)
\]
\[
c_t = f_t c_{t-1} + i_t \tanh \left( W_{xc} x_t + W_{hc} h_{t-1} + b_c \right)
\]
\[
o_t = \sigma \left( W_{xo} x_t + W_{ho} h_{t-1} + W_{co} c_t + b_o \right)
\]
\[
h_t = o_t \tanh(c_t)
\]

Figure from (Graves et al., 2013)
Long Short-Term Memory (LSTM)
Deep Bidirectional LSTM (DBLSTM)

Network training follows the standard approach used in hybrid systems [4]. Frame-level state targets are provided on the training set by a forced alignment given by a GMM-HMM system. The network is then trained to minimise the cross-entropy error of the targets using a softmax output layer with as many units as the total number of possible HMM states. At decoding time, the state probabilities yielded by the network are combined with a dictionary and language model to determine the most probable transcription. For a length $T$ acoustic sequence $x$, the network produces a length $T$ output sequence $y$, where each $y_t$ defines a probability distribution over the $K$ possible states: that is, $y_k(t)$ is the network’s estimate for the probability of observing state $k$ at time $t$ given $x$. Given a length $T$ state target sequence $z$, the network is trained to minimise the negative log-probability of the target sequence given the input sequence:

$$\log \Pr(z|x) = \sum_{t=1}^{T} \log y_{z(t)}(t) \tag{13}$$

Which leads to the following error derivatives at the output layer:

$$\frac{\partial \log \Pr(z|x)}{\partial \hat{y}_k(t)} = y_k(t) \tag{14}$$

where $\hat{y}_t$ is the vector of output activations before they have been normalised with the softmax function. These derivatives are then fed back through the network using backpropagation through time to determine the weight gradient.

When training deep networks in hybrid systems with stochastic gradient descent it has been found advantageous to select minibatches of frames randomly from the whole training set, rather than using whole utterances as batches. This is impossible with RNN-HMM hybrids because the weight gradients are a function of the entire utterance. Another difference is that hybrid deep networks are trained with an acoustic context window of frames to either side of the one being classified. This is not necessary for DBLSTM, since it is as able to store past and future context internally, and the data was therefore presented a single frame at a time.

For some of the experiments Gaussian noise was added to the network weights during training [15]. The noise was added once per training sequence, rather than at every timestep. Weight noise tends to ‘simplify’ neural networks, in the sense of reducing the amount of information required to transmit the parameters [16, 17], which improves generalisation.

4. TIMIT EXPERIMENTS

The first set of experiments were carried out on the TIMIT [18] speech corpus. Their purpose was to see how hybrid training for deep bidirectional LSTM compared with the end-to-end training methods described in [1]. To this end, we ensured that the data preparation, network architecture and training parameters were consistent with those in the previous work. To allow us to test for significance, we also carried out repeated runs of the previous experiments (which were only run once in the original paper). In addition, we ran hybrid experiments using a deep bidirectional RNN with $\text{tanh}$ hidden units instead of LSTM.

The standard 462 speaker set with all SA records removed was used for training, and a separate development set of 50 speakers was used for early stopping. Results are reported for the 24-speaker core test set. The audio data was preprocessed using a Fourier-transform-based filterbank with 40 coefficients (plus energy) distributed on a mel-scale, together with their first and second temporal derivatives. Each input

---

Figure from (Graves et al., 2013)
Deep Bidirectional LSTM (DBLSTM)

How important is this particular architecture?

Jozefowicz et al. (2015) evaluated 10,000 different LSTM-like architectures and found several variants that worked just as well on several tasks.
Mini-Batch SGD

• Gradient Descent:
  Compute true gradient exactly from all N examples

• Stochastic Gradient Descent (SGD):
  Approximate true gradient by the gradient of one randomly chosen example

• Mini-Batch SGD:
  Approximate true gradient by the average gradient of K randomly chosen examples
Mini-Batch SGD

while not converged: $\theta \leftarrow \theta - \lambda g$

Three variants of first-order optimization:

Gradient Descent: $g = \nabla J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \nabla J^{(i)}(\theta)$

SGD: $g = \nabla J^{(i)}(\theta)$ where $i$ sampled uniformly

Mini-batch SGD: $g = \frac{1}{S} \sum_{s=1}^{S} \nabla J^{(i_s)}(\theta)$ where $i_s$ sampled uniformly $\forall s$
RNN Training Tricks

• Deep Learning models tend to consist largely of matrix multiplications

• Training tricks:
  – mini-batching with masking
  – sorting into buckets of similar-length sequences, so that mini-batches have same length sentences
  – truncated BPTT, when sequences are too long, divide sequences into chunks and use the final vector of the previous chunk as the initial vector for the next chunk (but don’t backprop from next chunk to previous chunk)

<table>
<thead>
<tr>
<th>Metric</th>
<th>DyC++</th>
<th>DyPy</th>
<th>Chainer</th>
<th>DyC++ Seq</th>
<th>Theano</th>
<th>TF</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNNLM (MB=1)</td>
<td>words/sec</td>
<td>190</td>
<td>190</td>
<td>114</td>
<td>494</td>
<td>189</td>
</tr>
<tr>
<td>RNNLM (MB=4)</td>
<td>words/sec</td>
<td>830</td>
<td>825</td>
<td>295</td>
<td>1510</td>
<td>567</td>
</tr>
<tr>
<td>RNNLM (MB=16)</td>
<td>words/sec</td>
<td>1820</td>
<td>1880</td>
<td>794</td>
<td>2400</td>
<td>1100</td>
</tr>
<tr>
<td>RNNLM (MB=64)</td>
<td>words/sec</td>
<td>2440</td>
<td>2470</td>
<td>1340</td>
<td>2820</td>
<td>1260</td>
</tr>
</tbody>
</table>

Table from Neubig et al. (2017)
RNN Summary

• RNNs
  – Applicable to tasks such as sequence labeling, speech recognition, machine translation, etc.
  – Able to learn context features for time series data
  – Vanishing gradients are still a problem – but LSTM units can help

• Other Resources
  – Christopher Olah’s blog post on LSTMs
    http://colah.github.io/posts/2015-08-Understanding-LSTMs/
RNN LANGUAGE MODELS
Two Key Ingredients

## Language Models

| context | target | $P(w_t|w_{t-1}, w_{t-2}, \ldots w_{t-5})$ |
|---------|--------|----------------------------------|
| the cat sat on the | mat | 0.15 |
| $w_{t-5}$ $w_{t-4}$ $w_{t-3}$ $w_{t-2}$ $w_{t-1}$ | $w_t$ | 0.12 |
| the cat sat on the | rug | 0.09 |
| the cat sat on the | hat | 0.01 |
| the cat sat on the | dog | 0 |
| the cat sat on the | the | 0 |
| the cat sat on the | sat | 0 |
| the cat sat on the | robot | ? |
| the cat sat on the | printer | ? |
n-grams

\[ m_{w,c} \propto \frac{\#(w,c)}{\#(c)} \]

- the cat sat on the mat
- the cat drinks milk
- the dog chases the cat
- the paws of the cat
- the cat chases the rat
- the rat eats cheese
- the rat eats the mat

Slide Credit: Piotr Mirowski

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
n-grams

\[ P(w_1, w_2, \ldots, w_{T-1}, w_T) \approx \prod_{t=1}^{T} P(w_t|w_{t-1}, \ldots, w_{t-n+1}) \]

Slide Credit: Piotr Mirowski

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
The Chain Rule

\[ P(w_1, w_2, \ldots, w_{T-1}, w_T) = \prod_{t=1}^{T} P(w_t|w_{t-1}, w_{t-2}, \ldots, w_1) \]

The Chain Rule

\[ P(w_1) \]
\[ P(w_2|w_1) \]
\[ P(w_3|w_2, w_1) \]
\[ P(w_4|w_3, w_2, w_1) \]
\[ P(w_5|w_4, w_3, w_2, w_1) \]
\[ P(w_6|w_5, w_4, w_3, w_2, w_1) \]
A Key Insight: vectorizing context


\[ p(w_t | w_1, \ldots, w_{t-1}) = p_\theta(w_t | f_\theta(w_1, \ldots, w_{t-1})) \]

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Recurrent Neural Network Language Models


“persistent memory”: state variable for arbitrarily long contexts

\[ z_t = \tanh(Wz_{t-1} + Uw_t) \]
\[ p(w_{t+1}) = \text{softmax}(Bz_t) \]
Recurrent Neural Network Language Models

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Recurrent Neural Network Language Models
Recurrent Neural Network Language Models
Recurrent Neural Network Language Models

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
What do we Optimize?

$$\theta^* = \arg \max_\theta E_{w \sim \text{data}} \log P_\theta(w_1, \ldots, w_T)$$
Recurrent Neural Network Language Models

Learning Sequences — Piotr Mirowski

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Recurrent Neural Network Language Models

Slide from Vinyals & Jaitly (ICML Tutorial, 2017)
Sampling from an RNN-LM

Shakespeare’s As You Like It

VIOLA: Why, Salisbury must find his flesh and thought That which I am not aps, not a man and in fire, To show the reining of the raven and the wars To grace my hand reproach within, and not a fair are hand, That Caesar and my goodly father's world; When I was heaven of presence and our fleets, We spare with hours, but cut thy council I am great, Murdered and by thy master's ready there My power to give thee but so much as hell: Some service in the noble bondman here, Would show him to her wine.

KING LEAR: O, if you were a feeble sight, the courtesy of your law, Your sight and several breath, will wear the gods With his heads, and my hands are wonder'd at the deeds, So drop upon your lordship's head, and your opinion Shall be against your honour.

RNN-LM Sample

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Example from http://karpathy.github.io/2015/05/21/rnn-effectiveness/
Sampling from an RNN-LM

RNN-LM Sample

VIOLA: Why, Salisbury must find his flesh and thought That which I am not aps, not a man and in fire, To show the reining of the raven and the wars To grace my hand reproach within, and not a fair are hand, That Caesar and my goodly father's world; When I was heaven of presence and our fleets, We spare with hours, but cut thy council I am great, Murdered and by thy master's ready there My power to give thee but so much as hell: Some service in the noble bondman here, Would show him to her wine.

KING LEAR: O, if you were a feeble sight, the courtesy of your law, Your sight and several breath, will wear the gods With his heads, and my hands are wonder'd at the deeds, So drop upon your lordship's head, and your opinion Shall be against your honour.

Shakespeare’s As You Like It

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.
Sampling from an RNN-LM

VIOLA: Why, Salisbury must find his flesh and thought That which I am not aps, not a man and in fire, To show the reining of the raven and the wars To grace my hand reproach within, and not a fair are hand, That Caesar and my goodly father's world; When I was heaven of presence and our fleets, We spare with hours, but cut thy council I am great, Murdered and by thy master's ready there My power to give thee but so much as hell: Some service in the noble bondman here, Would show him to her wine.

KING LEAR: O, if you were a feeble sight, the courtesy of your law, Your sight and several breath, will wear the gods With his heads, and my hands are wonder'd at the deeds, So drop upon your lordship's head, and your opinion Shall be against your honour.

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him his but young and tender; and, I would be loath to foil him, as I honour, if he come in: by love to you, I came hither to acquaint you with, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.
Language Modeling

An aside:

• State-of-the-art language models currently tend to rely on transformer networks (e.g. GPT-2)

• RNN-LMs comprised most of the early neural LMs that led to current SOTA architectures

Figure from https://paperswithcode.com/sota/language-modelling-on-penn-treebank-word
RNN Language Models

Whiteboard:

– RNNLM for scoring of a path in a search space
– What’s missing? Dependence on the input.
SEQUENCE-TO-SEQUENCE MODELS
Sequence-to-Sequence Models

Motivating Question:
How can we model input/output pairs when the length of the input might be different from the length of the output?
Sequence-to-Sequence Models

Whiteboard:

– encoder-decoder architectures
– Example: biLSTM + RNNLM
Learning to Search for seq2seq

Whiteboard:

– DAgger for seq2seq
– Scheduled Sampling (a special case of DAgger)
Teacher Forcing
Teacher Forcing is the **supervised approach to imitation** when used to train RNNs

**Algorithm:**
1. feed the **ground truth** from the previous time step in as the input to the next time step
2. at each timestep **minimize cross entropy** (or some loss) of the **ground truth** for that time step

Scheduled Sampling
Scheduled Sampling is **online DAgger** with a variety of schedules for mixing the oracle policy and model policy when used to train RNNs

**Algorithm:**
1. feed the **model’s prediction (or with some probability the ground truth)** from the previous time step in as the input to the next time step
2. at each timestep **minimize cross entropy** (or some loss) of the **ground truth** for that time step
3. **gradually decrease the probability of** feeding in the **ground truth** with each iteration of training
L2S in deep-learning-speak

Scheduled Sampling

Scheduled Sampling is online DAgger with a variety of schedules for mixing the oracle policy and model policy when used to train RNNs

Algorithm:
1. feed the model’s prediction (or with some probability the ground truth) from the previous time step in as the input to the next time step
2. at each timestep minimize cross entropy (or some loss) of the ground truth for that time step
3. gradually decrease the probability of feeding in the ground truth with each iteration of training

Figure 2: Examples of decay schedules.