“The Malfoys!” said Hermione.

Harry was watching him. He looked like Madame Maxime. When she strode up the wrong staircase to visit himself.

“I’m afraid I’ve definitely been suspended from power, no chance—indeed?” said Snape. He put his head back behind them and read groups as they crossed a corner and fluttered down onto their ink lamp, and picked up his spoon. The doorbell rang. It was a lot cleaner down in London.
**Conditioned Language Models**

- Not just generate text, generate text according to some specification

<table>
<thead>
<tr>
<th>Input $X$</th>
<th>Output $Y$ (Text)</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structured Data</td>
<td>NL Description</td>
<td>NL Generation</td>
</tr>
<tr>
<td>English Document</td>
<td>Japanese Short Description</td>
<td>Translation</td>
</tr>
<tr>
<td>Utterance</td>
<td>Response</td>
<td>Summarization</td>
</tr>
<tr>
<td>Image</td>
<td>Text</td>
<td>Response Generation</td>
</tr>
<tr>
<td>Speech</td>
<td>Transcript</td>
<td>Image Captioning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Speech Recognition</td>
</tr>
</tbody>
</table>
Formulation and Modeling
Calculating the Probability of a Sentence

\[ P(X) = \prod_{i=1}^{I} P(x_i \mid x_1, \ldots, x_{i-1}) \]
Conditional Language Models

\[ P(Y|X) = \prod_{j=1}^{J} P(y_j | X, y_1, \ldots, y_{j-1}) \]
(One Type of) Language Model (Mikolov et al. 2011)

LSTM

predict

I

hate

this

movie

<s>

</s>
(One Type of) Conditional Language Model
(Sutskever et al. 2014)

Encoder

Decoder
How to Pass Hidden State?

- Initialize decoder w/ encoder (Sutskever et al. 2014)

- Transform (can be different dimensions)

- Input at every time step (Kalchbrenner & Blunsom 2013)
Methods of Generation
The Generation Problem

• We have a model of $P(Y|X)$, how do we use it to generate a sentence?

• Two methods:

  • **Sampling**: Try to generate a *random* sentence according to the probability distribution.

  • **Argmax**: Try to generate the sentence with the *highest* probability.
Ancestral Sampling

- **Randomly generate** words one-by-one.

  \[
  \text{while } y_{j-1} \neq "<\text{s}>": \\
  y_j \sim P(y_j | X, y_1, \ldots, y_{j-1})
  \]

- **An exact method** for sampling from $P(X)$, no further work needed.
Greedy Search

• One by one, pick the single highest-probability word

\[
\text{while } y_{j-1} \neq "</s>": \\
y_j = \arg\max P(y_j \mid X, y_1, \ldots, y_{j-1})
\]

• Not exact, real problems:
  
  • Will often generate the “easy” words first
  
  • Will prefer multiple common words to one rare word
Beam Search

• Instead of picking one high-probability word, maintain several paths

• Some in reading materials, more in a later class
Let’s Try it Out!
enc_dec.py
Model Ensembling
Ensembling

• Combine predictions from multiple models

• Why?
  • Multiple models make somewhat uncorrelated errors
  • Models tend to be more uncertain when they are about to make errors
  • Smooths over idiosyncrasies of the model
Linear Interpolation

- Take a weighted average of the M model probabilities

\[ P(y_j \mid X, y_1, \ldots, y_{j-1}) = \sum_{m=1}^{M} P_m(y_j \mid X, y_1, \ldots, y_{j-1}) P(m \mid X, y_1, \ldots, y_{j-1}) \]

- Second term often set to uniform distribution 1/M

- Probability according to model \( m \)

- Probability of model \( m \)
Log-linear Interpolation

- Weighted combination of log probabilities, normalize

\[ P(y_j \mid X, y_1, \ldots, y_{j-1}) = \]

\[
\text{softmax} \left( \sum_{m=1}^{M} \lambda_m (X, y_1, \ldots, y_{j-1}) \log P_m(y_j \mid X, y_1, \ldots, y_{j-1}) \right)
\]

- Interpolation coefficient often set to uniform distribution \( 1/M \)
Linear or Log Linear?

- Think of it in logic!

- **Linear**: “Logical OR”
  - the interpolated model likes any choice that a model gives a high probability
  - use models with models that capture different traits
  - necessary when any model can assign zero probability

- **Log Linear**: “Logical AND”
  - interpolated model only likes choices where all models agree
  - use when you want to restrict possible answers
Parameter Averaging

- **Problem**: Ensembling means we have to use $M$ models at test time, increasing our time/memory complexity.

- Parameter averaging is a cheap way to get some good effects of ensembling.

- Basically, write out models several times near the end of training, and take the average of parameters.
Ensemble Distillation (e.g. Kim et al. 2016)

- **Problem:** parameter averaging only works for models within the same run

- Knowledge distillation trains a model to **copy the ensemble**
  - Specifically, it tries to match the description over predicted words
  - Why? We want the model to make the same mistakes as an ensemble
  - Shown to increase accuracy notably
Stacking

• What if we have two very different models where prediction of outputs is done in very different ways?

• e.g. a word-by-word translation model and character-by-character translation model

• Stacking uses the output of one system in calculating features for another system
How do we Evaluate?
Basic Evaluation Paradigm

• Use parallel test set
• Use system to generate translations
• Compare target translations w/ reference
Human Evaluation

• Ask a human to do evaluation

太郎が花子を訪れた
Taro visited Hanako  the Taro visited the Hanako  Hanako visited Taro

<table>
<thead>
<tr>
<th>Adequate?</th>
<th>Fluent?</th>
<th>Better?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>1</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>2</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>3</td>
</tr>
</tbody>
</table>

• Final goal, but slow, expensive, and sometimes inconsistent
BLEU

- Works by comparing n-gram overlap with reference

Reference: Taro visited Hanako
System: the Taro visited the Hanako

Brevity: \( \min(1, \frac{|\text{System}|}{|\text{Reference}|}) = \min(1, \frac{5}{3}) \)  
\( \text{brevity penalty} = 1.0 \)

\[ \text{BLEU-2} = \left( \frac{3}{5} \times \frac{1}{4} \right)^{1/2} \times 1.0 \]
\[ = 0.387 \]

- **Pros**: Easy to use, good for measuring system improvement

- **Cons**: Often doesn’t match human eval, bad for comparing very different systems
METEOR

• Like BLEU in overall principle, with many other tricks: consider paraphrases, reordering, and function word/content word difference

• **Pros:** Generally significantly better than BLEU, esp. for high-resource languages

• **Cons:** Requires extra resources for new languages (although these can be made automatically), and more complicated
Perplexity

- Calculate the perplexity of the words in the held-out set *without* doing generation

- **Pros**: Naturally solves multiple-reference problem!

- **Cons**: Doesn’t consider decoding or actually generating output.

- May be reasonable for problems with lots of ambiguity.
What Do We Condition On?
From Structured Data
(e.g. Wen et al 2015)

- When you say “Natural Language Generation” to an old-school NLPer, it means this

<table>
<thead>
<tr>
<th>act type</th>
<th>SF Restaurant</th>
<th>SF Hotel</th>
</tr>
</thead>
<tbody>
<tr>
<td>inform, inform_only, reject, confirm, select, request, reqmore, goodbye</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>shared</th>
<th>SF Restaurant</th>
<th>SF Hotel</th>
</tr>
</thead>
<tbody>
<tr>
<td>name, type, *pricerange, price, phone, address, postcode, *area, *near</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>specific</th>
<th>SF Restaurant</th>
<th>SF Hotel</th>
</tr>
</thead>
<tbody>
<tr>
<td>*food</td>
<td></td>
<td>*hasinternet</td>
</tr>
<tr>
<td>*goodformal</td>
<td></td>
<td>*acceptscards</td>
</tr>
<tr>
<td>*kids-allowed</td>
<td></td>
<td>*dogs-allowed</td>
</tr>
</tbody>
</table>

*bold=binary slots, *=slots can take “don’t care” value
From Input + Labels
(e.g. Zhou and Neubig 2017)

- For example, word + morphological tags -> inflected word

- Other options: politeness/gender in translation, etc.
From Images
(e.g. Karpathy et al. 2015)

• Input is image features, output is text

“A Tabby cat is leaning on a wooden table, with one paw on a laser mouse and the other on a black laptop”
Other Auxiliary Information

• Name of a recipe + ingredients -> recipe (Kiddon et al. 2016)

• TED talk description -> TED talk (Hoang et al. 2016)

• etc. etc.
Questions?