Administrivia

- A4 and A5 grading underway
- Final project check-ins due in one week

Today

- Rest of the course: applications
  - Knowledge base QA
  - Open retrieval QA
- Next two lectures: generation and dialogue systems

Previously: SQuAD

Q: What was Marie Curie the first female recipient of?

Passage: One of the most famous people born in Warsaw was Marie Skłodowska-Curie, who achieved international recognition for her research on radioactivity and was the first female recipient of the **Nobel Prize**. Famous musicians include Władysław Szpilman and Frédéric Chopin. Though Chopin was born in the village of Żelazowa Wola, about 60 km (37 mi) from Warsaw, he moved to the city with his family when he was seven months old. Casimir Pulaski, a Polish general and hero of the American Revolutionary War, was born here in 1745.

Answer = Nobel Prize

- **Assume we know a passage that contains the answer**

Types of QA

- *What were the main causes of World War II?* — requires summarization
- *Can you get the flu from a flu shot?* — want IR to provide an explanation of the answer, not just yes/no
- *What was Marie Curie the first female recipient of?* — could be written down in a KB but probably isn't
- *How long should I soak dry pinto beans?*
- *When as Marie Curie born?* — we should just find this in a knowledge base
- Today: QA when it requires retrieving the answer from a passage

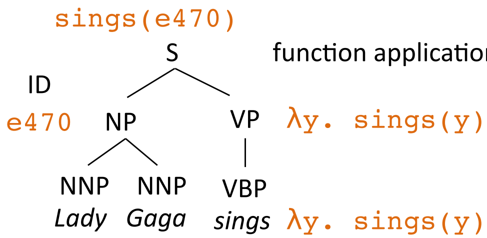
QA Pipeline

Semantic Parsing

Logical Forms I

Logical Forms II

Montague Semantics



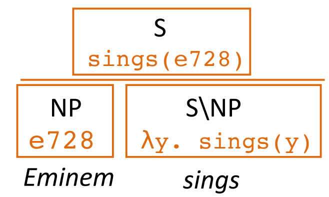
function application: apply this to e470

takes one argument (y, the entity) and returns a logical form `sings(y)`

- We can use the syntactic parse as a bridge to the lambda-calculus representation, build up a logical form (our model) *compositionally*

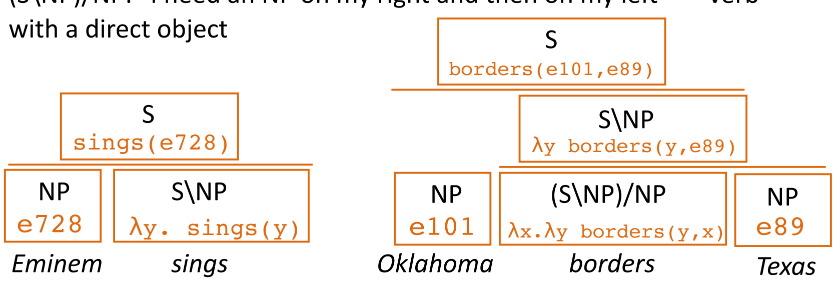
Combinatory Categorial Grammar

- Steedman+Szabolcsi (1980s): formalism bridging syntax and semantics
- Parallel derivations of syntactic parse and lambda calculus expression
- Syntactic categories (for this lecture): S, NP, "slash" categories
- S\NP: "if I combine with an NP on my left side, I form a sentence" — verb
- When you apply this, there has to be a parallel instance of function application on the semantics side



Combinatory Categorial Grammar

- Steedman+Szabolcsi (1980s): formalism bridging syntax and semantics
- Syntactic categories (for this lecture): S, NP, "slash" categories
  - S\NP: "if I combine with an NP on my left side, I form a sentence" — verb
  - (S\NP)/NP: "I need an NP on my right and then on my left" — verb with a direct object



CCG Parsing

$$
\begin{array}{cccc}
\text{What} & \text{states} & \text{border} & \text{Texas} \\
\hline
(S/(S\backslash NP))/N & N & (S\backslash NP)/NP & NP \\
\lambda f.\lambda g.\lambda x.f(x) \wedge g(x) & \lambda x.state(x) & \lambda x.\lambda y.borders(y,x) & texas \\
 & & \multicolumn{2}{c}{\overline{\qquad\qquad\qquad\qquad}{>}} \\
 & & \multicolumn{2}{c}{(S\backslash NP)} \\
 & & \multicolumn{2}{c}{\lambda y.borders(y, texas)}
\end{array}
$$

- "What" is a ***very*** complex type: needs a noun and needs a S\NP to form a sentence. S\NP is basically a verb phrase (*border Texas*)

Zettlemoyer and Collins (2005)

CCG Parsing

$$
\begin{array}{cccc}
\text{What} & \text{states} & \text{border} & \text{Texas} \\
\hline
(S/(S\backslash NP))/N & N & (S\backslash NP)/NP & NP \\
\lambda f.\lambda g.\lambda x.f(x) \land g(x) & \lambda x.state(x) & \lambda x.\lambda y.borders(y,x) & texas
\end{array}
$$

$$
\frac{}{S/(S\backslash NP) \quad \lambda g.\lambda x.state(x) \land g(x)} > \qquad \frac{}{(S\backslash NP) \quad \lambda y.borders(y, texas)} >
$$

$$
\frac{}{S \quad \lambda x.state(x) \land borders(x, texas)} >
$$

- "What" is a ***very*** complex type: needs a noun and needs a S\NP to form a sentence. S\NP is basically a verb phrase (*border Texas*)
- *What* in this case knows that there are two predicates (*states* and *border Texas*). This is not a general thing

Zettlemoyer and Collins (2005)

CCG Parsing

- These question are *compositional*: we can build bigger ones out of smaller pieces

What states border Texas?

What states border states bordering Texas?

What states border states bordering states bordering Texas?

CCG Parsing

- Many ways to build these parsers
- One approach: run a "supertagger" (tags the sentence with complex labels), then run the parser

$$
\begin{array}{cccc}
\text{What} & \text{states} & \text{border} & \text{Texas} \\
\hline
(S/(S\backslash NP))/N & N & (S\backslash NP)/NP & NP \\
\lambda f.\lambda g.\lambda x.f(x) \land g(x) & \lambda x.state(x) & \lambda x.\lambda y.borders(y,x) & texas
\end{array}
$$

  - Parsing is easy once you have the tags, so we've reduced it to a (hard) tagging problem

Zettlemoyer and Collins (2005)

Training CCG Parsers

- Training data looks like pairs of sentences and logical forms

What states border Texas	`λx. state(x) ∧ borders(x, e89)`
What borders Texas	`λx. borders(x, e89)`
...	

- Unlike PCFGs, we don't know which words yielded which fragments of CCG
- Requires an "unsupervised" approach like Model 1 for word alignment

Zettlemoyer and Collins (2005)

Seq2seq Semantic Parsing

Semantic Parsing as Translation

"what states border Texas"

↓

```
lambda x ( state ( x ) and border ( x , e89 ) ) )
```

- Write down a linearized form of the semantic parse, train seq2seq models to directly translate into this representation
- What are some benefits of this approach compared to grammar-based?
- What might be some concerns about this approach? How do we mitigate them?

Jia and Liang (2016)

Handling Invariances

"what states border Texas" *"what states border Ohio"*

- Parsing-based approaches handle these the same way
  - Possible divergences: features, different weights in the lexicon
- Can we get seq2seq semantic parsers to handle these the same way?
- Key idea: do data augmentation by synthetically creating more data from a single example

Semantic Parsing as Translation

GEO

x: *"what is the population of iowa ?"*

```
y: _answer ( NV , (
  _population ( NV , V1 ) , _const (
    V0 , _stateid ( iowa ) ) ) )
```

ATIS

x: *"can you list all flights from chicago to milwaukee"*

```
y: ( _lambda $0 e ( _and
  ( _flight $0 )
  ( _from $0 chicago : _ci )
  ( _to $0 milwaukee : _ci ) ) )
```

Overnight

x: *"when is the weekly standup"*

```
y: ( call listValue ( call
    getProperty meeting.weekly_standup
    ( string start_time ) ) )
```

- Prolog
- Lambda calculus
- Other DSLs

- Handle all of these with uniform machinery!

Jia and Liang (2016)

Semantic Parsing as Translation

	Geo	ATIS
Previous Work		
Zettlemoyer and Collins (2007)		**84.6**
Kwiatkowski et al. (2010)	88.9	
Liang et al. (2011)[2]	91.1	
Kwiatkowski et al. (2011)	88.6	82.8
Poon (2013)		83.5
Zhao and Huang (2015)	88.9	84.2
Our Model		
No Recombination	85.0	76.3
AbsEntities	85.4	79.9
AbsWholePhrases	87.5	
Concat-2	84.6	79.0
Concat-3		77.5
AWP + AE	88.9	
AE + C2		78.8
AWP + AE + C2	**89.3**	
AE + C3		83.3

- Three forms of data augmentation all help
- Results on these tasks are still not as strong as hand-tuned systems from 10 years ago, but the same simple model can do well at all problems

Jia and Liang (2016)

Applications

- GeoQuery (Zelle and Mooney, 1996): answering questions about states (~80% accuracy)
- Jobs: answering questions about job postings (~80% accuracy)
- ATIS: flight search
- Can do well on all of these tasks if you handcraft systems and use plenty of training data: these domains aren't that rich

Retrieval Models

Types of QA

How long should I soak dry pinto beans?

↓ execute search (*retrieval*)

https://minimalistbaker.com › mexican-pinto-beans-scratc...

Easy Pinto Beans From Scratch (1-Pot!) - Minimalist Baker

How Long to Soak Pinto Beans

We have found that 6-8 hours is the optimal amount of time for soaking dry pinto beans. The longer you soak them, the more tender they will become, and the more likely they will split and separate during cooking.

↓ show snippet to user (*answer extraction*)

We have found that **6-8 hours** is the optimal amount of time for soaking dry pinto beans. The longer you soak them, the more tender they will become, and the more likely they will split and separate during cooking. So if you can't get to them right away, simply drain, cover, and refrigerate until ready to use.

Open-domain QA

- SQuAD-style QA from a paragraph is very artificial, not a real application
- Real QA systems should be able to handle more than just a paragraph of context — theoretically should work over the whole web?

Q: *What was Marie Curie the recipient of?*

Marie Curie was awarded the Nobel Prize in Chemistry and the Nobel Prize in Physics…

Mother Teresa received the Nobel Peace Prize in…

Curie received his doctorate in March 1895…

Skłodowska received accolades for her early work…

Open-domain QA

- SQuAD-style QA from a paragraph is very artificial, not a real application
- Real QA systems should be able to handle more than just a paragraph of context — theoretically should work *open-domain* over the whole web
- **Open-domain QA** pipeline: given a question:
  - Retrieve some documents with an IR system, usually either classic IR (tf-idf, indexed documents) or dense neural system
  - Zero in on the answer in those documents with a QA model — this part looks very similar to SQuAD

DrQA

- Uses Lucene, basically sparse tf-idf vectors. How often does the retrieved context contain the answer?
- Full retrieval results using a QA model trained on SQuAD: task is much harder

Dataset	Wiki Search	Doc. Retriever plain	Doc. Retriever +bigrams
SQuAD	62.7	76.1	**77.8**
CuratedTREC	81.0	85.2	**86.0**
WebQuestions	73.7	**75.5**	74.4
WikiMovies	61.7	54.4	**70.3**

Dataset	SQuAD
SQuAD *(All Wikipedia)*	27.1
CuratedTREC	19.7
WebQuestions	11.8
WikiMovies	24.5

Chen et al. (2017)

Problems

- Many SQuAD questions are not suited to the "open" setting because they're underspecified

  Where did the Super Bowl take place?

  Which player on the Carolina Panthers was named MVP?

- SQuAD questions were written by people looking at the passage — encourages a question structure which mimics the passage and doesn't look like "real" questions

Lee et al. (2019)

NaturalQuestions Dataset

- Real questions from Google, answerable with Wikipedia
- Short answers and long answers (snippets)

Question:

where is blood pumped after it leaves the right ventricle?

Short Answer:

None

Long Answer:

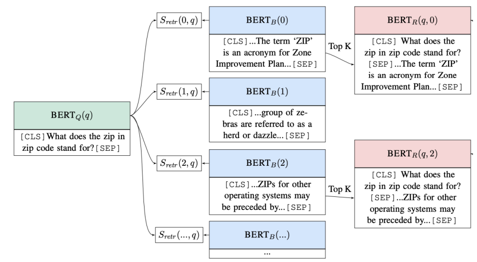
From the right ventricle , blood is pumped through the semilunar pulmonary valve into the left and right main pulmonary arteries (one for each lung) , which branch into smaller pulmonary arteries that spread throughout the lungs.

- Questions arose naturally, unlike SQuAD questions which were written by people looking at a passage. This makes them much harder

Kwiatkowski et al. (2019)

Retrieval with BERT

- Can we do better than a simple IR system?
- Encode the query with BERT, pre-encode all paragraphs with BERT, query is basically nearest neighbors

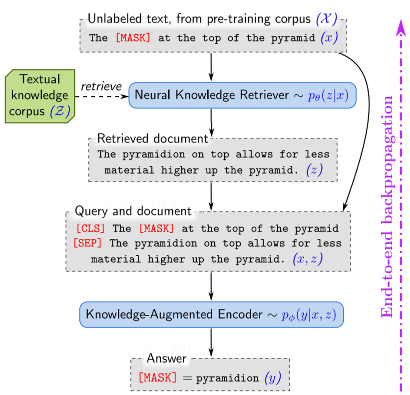


$$h_q = \mathbf{W_q}\mathrm{BERT}_Q(q)[\mathtt{CLS}]$$
$$h_b = \mathbf{W_b}\mathrm{BERT}_B(b)[\mathtt{CLS}]$$
$$S_{retr}(b,q) = h_q^\top h_b$$

Lee et al. (2019)

REALM

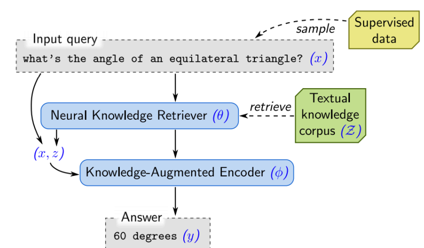
- Technique for integrating retrieval into pre-training
- Retriever relies on a maximum inner-product search (MIPS) over BERT embeddings
- MIPS is fast — challenge is how to refresh the BERT embeddings



Guu et al. (2020)

REALM

- Fine-tuning can exploit the same kind of textual knowledge
- Can work for tasks requiring knowledge lookups



Guu et al. (2020)

REALM

Name	Architectures	Pre-training	NQ (79k/4k)	WQ (3k/2k)	CT (1k /1k)	# params
BERT-Baseline (Lee et al., 2019)	Sparse Retr.+Transformer	BERT	26.5	17.7	21.3	110m
T5 (base) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	27.0	29.1	-	223m
T5 (large) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	29.8	32.2	-	738m
T5 (11b) (Roberts et al., 2020)	Transformer Seq2Seq	T5 (Multitask)	34.5	37.4	-	11318m
DrQA (Chen et al., 2017)	Sparse Retr.+DocReader	N/A	-	20.7	25.7	34m
HardEM (Min et al., 2019a)	Sparse Retr.+Transformer	BERT	28.1	-	-	110m
GraphRetriever (Min et al., 2019b)	GraphRetriever+Transformer	BERT	31.8	31.6	-	110m
PathRetriever (Asai et al., 2019)	PathRetriever+Transformer	MLM	32.6	-	-	110m
ORQA (Lee et al., 2019)	Dense Retr.+Transformer	ICT+BERT	33.3	36.4	30.1	330m
Ours ($\mathcal{X}$ = Wikipedia, $\mathcal{Z}$ = Wikipedia)	Dense Retr.+Transformer	REALM	39.2	40.2	**46.8**	330m
Ours ($\mathcal{X}$ = CC-News, $\mathcal{Z}$ = Wikipedia)	Dense Retr.+Transformer	REALM	**40.4**	**40.7**	42.9	330m

- 330M parameters + a knowledge base beats an 11B parameter model

Guu et al. (2020)

Takeaways

- Two different types of QA presented here:
  - Knowledge base QA: parse the question into a logical form that you can execute against your knowledge base
  - Open-domain QA: what Google does; retrieves documents from the web, finds the answer there, and highlights it for you
- Next time: generative models