Giving BERT a Calculator: Finding Operations and Arguments with Reading Comprehension

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler — Google Research

Overview

The Task

- Question-Answering with a mix of classification, extractive, and semi-abstractive answers.
- Reasoning with numbers + arithmetic.
- Focus on DROP[1].

Our Approach

- Unify classification, span selection, and arithmetic reasoning as simple *program* generation.
- Programs are *operations*, whose *arguments* are pointers into the passage + other operations.
- Programs are scored using structured prediction.

Benefits

- We show that
  - Operations are expressive enough for multiple QA spaces.
  - One model performs well on many datasets.
  - It's easy to extend the operations, and learn them from a few hundred examples.
- Programs are somewhat interpretable.
- Programs are a natural way to interface with APIs.

A Tour Through the Model

Question: How many years after Killigrew's first wife died was it before he remarried?

Passage: Killigrew lived in Hanworth in Middlesex and Falmouth in Cornwall. On **4** November **1566** Killigrew married in the church of St Peter Le Poer, London, Catherine, fourth daughter of Sir Anthony Cooke. He thus became Cecil's brother-in-law. His wife died in **1583**, and on **7** November **1590** he was married in the same church to Jaél de Peigne, a French Hugenot. She was naturalised in June **1601**. After Henry's death she remarried on **19** April **1617** George Downham, Bishop of Derry, and died c.**1632**. By his first wife, Killigrew had four daughters: ...

① Enumerate candidate programs

{Span(his first wife), Diff(**1583**, **1590**), "7", Span(French), Span(**7**), "yes", Sum(**7**, **1566**), "unknown", "2", Span(c.**1632**), Merge(Falmouth, London), Diff100(**7**), Sum3(Sum(**1632**, **4**), **1590**), Span(married in the church of St Peter Le Poer), "no", Diff(**1632**, **1590**), Diff(**1583**, **1566**), "0", ...}

Up to ~48k candidates in a passage of length <512 wordpieces.

② Score Candidates with BERT and Beam Search

Program	Probability
Diff(**1583**, **1590**)	0.99
Diff(**1583**, **1566**)	<0.1
"yes"	≪1e-4
Span(his first wife)	≪1e-4
"7"	≪1e-4
...	...

Example scoring functions

$$\rho(\texttt{"yes"}) = \mathbf{w}_{\texttt{"yes"}}^{\intercal}\mathrm{MLP}_{\mathrm{lit}}(\mathbf{h}_{\mathrm{CLS}})$$

$$\rho(\texttt{Diff}, i, j) = \mathbf{w}_{\texttt{Diff}}^{\intercal}\mathrm{MLP}_{\mathrm{binary}}(\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_i \circ \mathbf{h}_j)$$

$$\rho(\texttt{Sum3}, (i, j), k) = \mathbf{w}_{\texttt{Sum3}}^{\intercal}\mathrm{MLP}_{\texttt{Sum3}}(\mathbf{h}_{ij}, \mathbf{h}_k) + \rho(\texttt{Sum}, i, j)$$

③ Argmax and Execute

Diff(**1583**, **1590**) → `str(abs(1583 - 1590))` → "7"

Answer: 7 ✅

Training with the Oracle

The task can be formulated as a latent variable problem:

$$q \xrightarrow{\text{Model}} z \xrightarrow{\text{Exec}} a$$

where the inference step

$$\text{Exec} : z \to a$$

is deterministic.

The *Oracle* generates all programs *z* that evaluate to answer *a*. Training maximizes the marginal likelihood over *all* Oracle programs:

$$\mathcal{J}(q, a) = -\log \sum_{z|a=\text{Exec}(z)} P(z|q)$$

$$P(z|q) = \frac{\exp \rho(z)}{\sum_{z' \in \text{Beam}} \exp \rho(z')}$$

Results on Multiple QA Datasets

Multitask

Default set of ops works with multiple QA datasets, including

- Extractive only: SQuAD[2], NQ[3]
- Classification only: BoolQ[4]
- A mix: DROP[1], CoQA[5]

	DROP F1	CoQA F1	BoolQ Accuracy
Our **multitask**	86.03	89.1	88.9
Our **separate**	86.15	89.4	88.0
SOTA	84.42[6]	91.3[7]	91*[8]
Human	96*	89.8	89*

Few-shot Learning New Ops

- Only a few hundred examples are needed to learn the Illinois Dataset (*N*=562), another style of QA setup with *new* `Mul` and `Div` ops.
- Model learns to use the previously unseen ops.
- Significant transfer learning from DROP.

There were 28 bales of hay in the barn. Tim stacked bales in the barn today. There are now 54 bales of hay in the barn. ***How many bales did he store in the barn?***

Accuracy (5x cross-validated)	
Our basic on Illinois data	48.6 ± 5.3
+ `Mul` and `Div` ops	74.0 ± 6.0
+ DROP data	83.2 ± 6.0
Deep RL[9]	73.3
SOTA[10]	80.1

Ablations on DROP

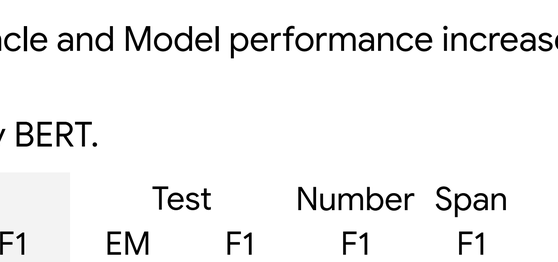
- As we add more operations, the Oracle and Model performance increase in equal measure.
- Co-training on CoQA helps, but only BERT.

	Oracle Dev EM	Dev EM	Dev F1	Test EM	Test F1	Number F1	Span F1
NAQANet[1]		46.8	50.4	44.2	47.8	45.0	64.8
Our basic BERT[11]	*80.0*	66.5	69.9			66.1	82.6
+`Diff100`	*88.8*	75.5	78.8			80.5	82.8
+`Sum3`	*90.2*	76.7	80.1			82.1	83.4
+`Merge`	*93.0*	77.0	80.5			82.1	83.4
+CoQA data	*93.0*	78.1	81.7	77.0	80.5	83.3	84.3
Our ALBERT[12] vers.		82.4	86.2	.	.	86.8	90.6
+CoQA data		82.5	86.1			86.8	90.4
+Ensemble		83.5	87.2			87.8	91.9

Spurious Ambiguities

In the example, three programs evaluate to the correct answer "7":

1. Diff(**1583**, **1590**)
2. Span(**7**)
3. "7"



- Such spurious ambiguity affects 40% of DROP numeric answers.
- During training, the model *learns to disambiguate*.
- Hard EM[13] makes even sparser predictions, but no better F1.

	Temp.	Entropy (bits)
Uniform	-	2.5
Unconstrained entropy	1	<0.2
Oracle-constrained	1	<0.1
Oracle-constrained	0.1	<0.05

Footnotes

* Test set results where Dev set results are not available/published.

[1] Dua et al. (2019) *DROP*
[2] Rajpurkar et al. (2018) *SQuAD 2*
[3] Kwiatkowski et al. (2019) *Natural Questions*
[4] Clark et al. (2019) *BoolQ*
[5] Reddy et al. (2019) *CoQA*
[6] As a proxy for SOTA, top of the DROP leaderboard as of 2019.11.1
[7] Ju et al. (2019) *Technical report on Conversational QA*
[8] Raffel et al. (2019) *Exploring the Limits of Transfer Learning*
[9] Wang et al. (2018) *MathDQN*
[10] Liang et al. (2016) *A tag-based statistical english math word problem solver*
[11] Devlin et al. (2019) *BERT*
[12] Lan et al. (2019) *ALBERT: A Lite BERT*
[13] Min et al. (2019) *A Discrete Hard EM Approach*