Table-based Fact Verification

彭凯龙

Introduction

United States House of Representatives Elections, 1972

District	Incumbent	Party	Result	Candidates
California 3	John E. Moss	democratic	re-elected	John E. Moss (d) 69.9% John Rakus (r) 30.1%
California 5	Phillip Burton	democratic	re-elected	Phillip Burton (d) 81.8% Edlo E. Powell (r) 18.2%
California 8	George Paul Miller	democratic	lost renomination democratic hold	Pete Stark (d) 52.9% Lew M. Warden , Jr. (r) 47.1%
California 14	Jerome R. Waldie	republican	re-elected	Jerome R. Waldie (d) 77.6% Floyd E. Sims (r) 22.4%
California 15	John J. Mcfall	republican	re-elected	John J. Mcfall (d) unopposed

Entailed Statement

1. John E. Moss and Phillip Burton are both re-elected in the house of representative election.
2. John J. Mcfall is unopposed during the re-election.
3. There are three different incumbents from democratic.

Refuted Statement

1. John E. Moss and George Paul Miller are both re-elected in the house of representative election.
2. John J. Mcfall failed to be re-elected though being unopposed.
3. There are five candidates in total, two of them are democrats and three of them are republicans.

Related work:

- Natural Language Inference & Reasoning
- Table Question Answering
- Program Synthesis & Semantic Parsing
- Fact Checking

TABFACT: A LARGE-SCALE DATASET FOR TABLE-BASED FACT VERIFICATION

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, William Yang Wang
University of California, Santa Barbara, CA, USA
Tencent AI Lab, Bellevue, WA, USA

TAB-FACT Verification Dataset

- Extract 16K tables from WikiTables with captions
- Manually annotated 118K statements classified as ENTAILED and REFUTED
- Less than 50 rows and 10 columns
- Overly complicated tables were filtered out (e.g. multirow, multicolumn, latex symbol)

Positive two-channel annotation:

- Simple channel:
  - corresponding to a single row/record in the table with unary fact
  - mention the cell values without dramatic modification or paraphrasing
- Complex channel:
  - involving multiple rows in the tables with higher-order semantics
  - rephrase the table records to involve more semantic understanding

Negative rewriting

rewrite the collected entailed statements
retain the sentence style/length to prevent artificial cues

TAB-FACT Verification Dataset

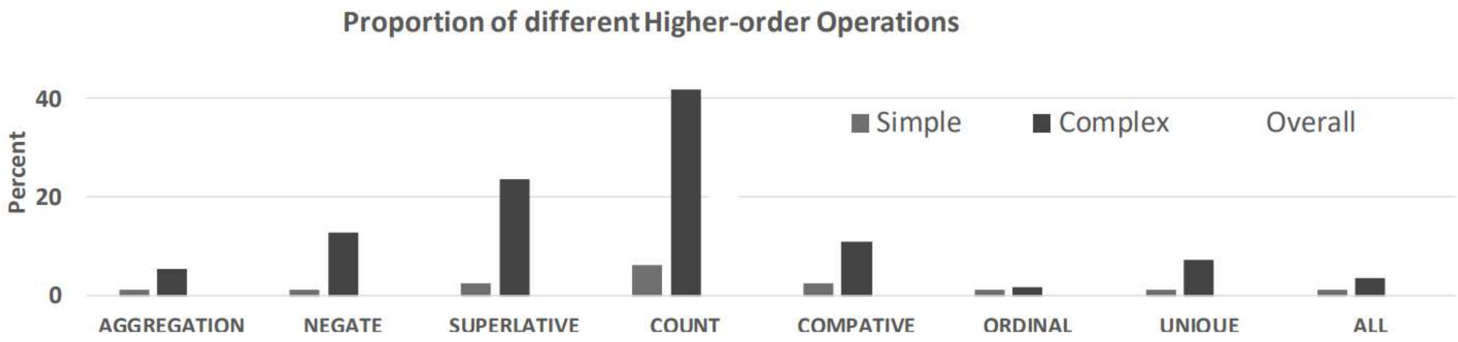


Figure 2: Proportion of different higher-order operations from the simple/complex channels.

Channel	#Sentence	#Table	Len(Ent)	Len(Ref)	Split	#Sentence	Table	Row	Col
Simple	50,244	9,189	13.2	13.1	Train	92,283	13,182	14.1	5.5
Complex	68,031	7,392	14.2	14.2	Val	12,792	1,696	14.0	5.4
Total	118,275	16,573	13.8	13.8	Test	12,779	1,695	14.2	5.4

Models

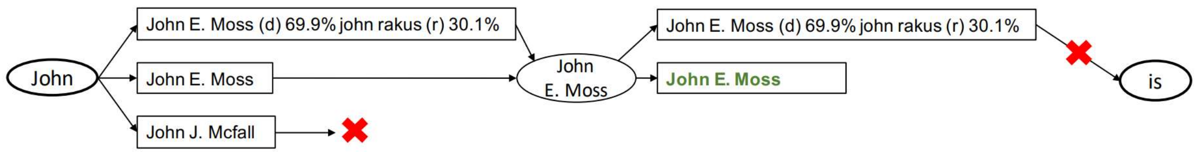
- Dataset $(\boldsymbol{T}, S, L)$: Table $\mathbf{T} = \{T_{i,j} | i \leq R_T, j \leq C_T\}$, Statement $S = s_1, \ldots, s_n$, Label $L \in \{0,1\}$
- Entity link

  longest string match

  minimum edit distance
- Do not feed the caption to the model

District	Incumbent	Party	Result	Candidates
California 3	John E. Moss	democratic	re-elected	John E. Moss (d) 69.9% John Rakus (r) 30.1%
California 5	Phillip Burton	democratic	re-elected	Phillip Burton (d) 81.8% Edlo E. Powell (r) 18.2%
california 8	George Paul Miller	democratic	lost renomination democratic hold	Pete Stark (d) 52.9% Lew M. Warden , Jr. (r) 47.1%
California 14	Jerome R. Waldie	republican	re-elected	Jerome R. Waldie (d) 77.6% Floyd E. Sims (r) 22.4%
California 15	John J. Mcfall	republican	re-elected	John J. Mcfall (d) unopposed

Statement: **John E. Moss** is a democratic who is from California 3 district



Two models:

- Latent Program Algorithm (LPA)
- TABLE-BERT

Latent Program Algorithm (LPA)

Formulate the table fact verification as a program synthesis problem
1. Latent program search
2. Discriminator ranking

1. Latent program search

parse the statement into programs

- define the plausible API set with 50 functions
- use trigger words to prune the API set

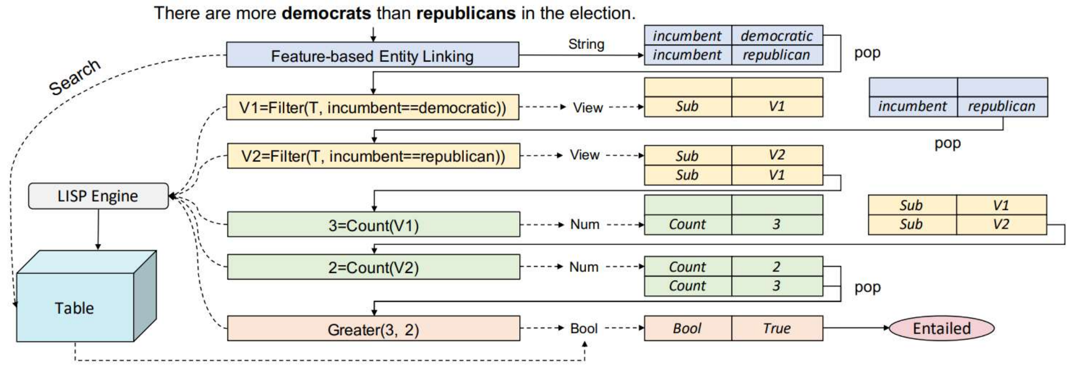
Trigger	Function
'average'	average
'difference', 'gap', 'than', 'separate'	diff
'sum', 'summation', 'combine', 'combined', 'total', 'add', 'all', 'there are'	ddd, sum
'not', 'no', 'never', "didn't", "won't", "wasn't", "isn't,"haven't", "weren't", "won't", 'neither', 'none', 'unable, 'fail', 'different', 'outside', 'unable', 'fail'	not_eq, not_within, Filter_not_eq, none
'not', 'no', 'none'	none
'first', 'top', 'latest', 'most'	first
'last', 'bottom', 'latest', 'most'	last
'RBR', 'JJR', 'more', 'than', 'above', 'after'	filter_greater, greater
'RBR', 'JJR', 'less', 'than', 'below', 'under'	filter_less, less
'all', 'every', 'each'	all_eq, all_less, all_greater,

Name	Arguments	Output
Count	View	Number
Within	View, Header String, Cell String/Number	Bool
Without	View, Header String, Cell String/Number	Bool
None	String	Bool
Before/After	Row, Row	Row
First/Second/Third/Fourth	View, Row	Bool
Average/Sum/Max/Min	View, Header String	Number
Argmin/ Argmax	View, Header String	Row
Hop	Row, Header String	Number/ String
Diff/Add	Number, Number	Number
Greater/Less	Number, Number	Bool
Equal/ Unequal	String, String/ Number, Number	Bool

Latent Program Algorithm (LPA)

Algorithm 1 Latent Program Search with Comments

Initialize Number Cache $\mathcal{N}$, String Cache $\mathcal{R}$, Bool Cache $\mathcal{B}$, View Cache $\mathcal{V} \rightarrow \emptyset$
Push linked numbers, strings from the given statement S into $\mathcal{N}, \mathcal{R}$, and push $\mathbf{T}$ into $\mathcal{V}$
Initialize the result collector $\mathcal{P} \rightarrow \emptyset$ and an empty program trace $P = \emptyset$
Initialize the Queue $\mathcal{Q} = [(P, \mathcal{N}, \mathcal{R}, \mathcal{B}, \mathcal{V})]$, we use $\mathcal{Q}$ to store the intermediate states
Use trigger words to find plausible function set $\mathcal{F}$, for example, $more$ will trigger $Greater$ function.
while loop over time $t = 1 \rightarrow$ MAXSTEP **do**:
    while $(P, \mathcal{N}, \mathcal{R}, \mathcal{B}, \mathcal{V}) = \mathcal{Q}.pop()$ **do**:
        while loop over function set $f \in \mathcal{F}$ **do**:
            if arguments of f are in the caches **then**
                Pop out the required arguments $arg_1, arg_2, \cdots, arg_n$ for different cachess.
                Execute $A = f(arg_1, \cdots, arg_n)$ and concatenate the program trace P.
                if Type(A)=Bool **then**
                    if $\mathcal{N} = \mathcal{S} = \mathcal{B} = \emptyset$ **then**
                        $\mathcal{P}.push((P, A))$ # The program P is valid since it consumes all the variables.
                        $P = \emptyset$ # Collect the valid program P into set $\mathcal{P}$ and reset P
                    else
                        $\mathcal{B}.push(A)$ # The intermediate boolean value is added to the bool cache
                        $\mathcal{Q}.push((P, \mathcal{N}, \mathcal{R}, \mathcal{B}, \mathcal{V}))$ # Add the refreshed state to the queue again
                if Type(A) $\in$ {Num, Str, View} **then**
                    if $\mathcal{N} = \mathcal{S} = \mathcal{B} = \emptyset$ **then**
                        $P = \emptyset$;break # The program ends without consuming the cache, throw it.
                    else
                        push A into $\mathcal{N}$ or $\mathcal{S}$ or $\mathcal{V}$ # Add the refreshed state to the queue for further search
                        $\mathcal{Q}.push((P, \mathcal{N}, \mathcal{R}, \mathcal{B}, \mathcal{V}))$
Return the triple $(\mathbf{T}, S, \mathcal{P})$ # Return (Table, Statement, Program Set)



Get potential program candidate

$$\mathcal{P} = \{(P_1, A_1), \cdots, (P_n, A_n)\}$$

Latent Program Algorithm (LPA)

2. Discriminator

- weakly supervised training algorithm:

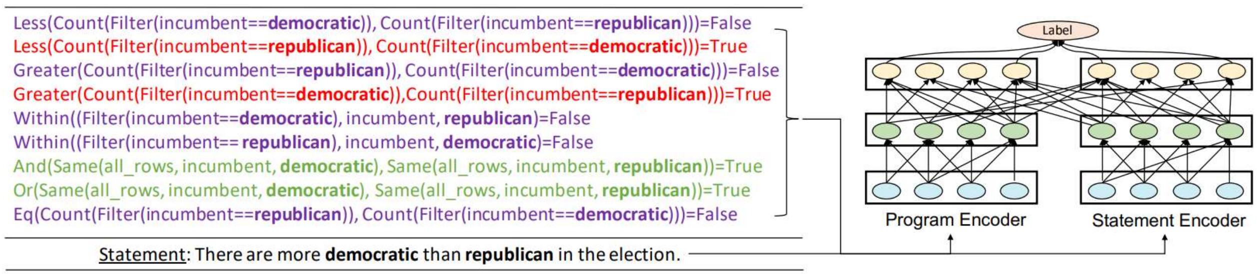
viewing all the label-consistent programs $\{P_i|(P_i, A_i); A_i = L\}$ as positive instances

Transformer-based encoder:

$$Enc^P(P) \in \mathbb{R}^{m \times D}$$

$$Enc^S(S) \in \mathbb{R}^{n \times D}$$

Concatenated both [CLS] output



Linear projection layer:

$$p_\theta(S, P) = \sigma(v_p^T[Enc^S(S); Enc^P(P)])$$

Aggregate with weights or rank the highest-confident

TABLE-BERT

Two-sequence binary classification problem

1. linearizing a table $\mathbf{T}$ into a sequence $\tilde{\mathbf{T}}$

only retaining the columns containing entities linked to the statement

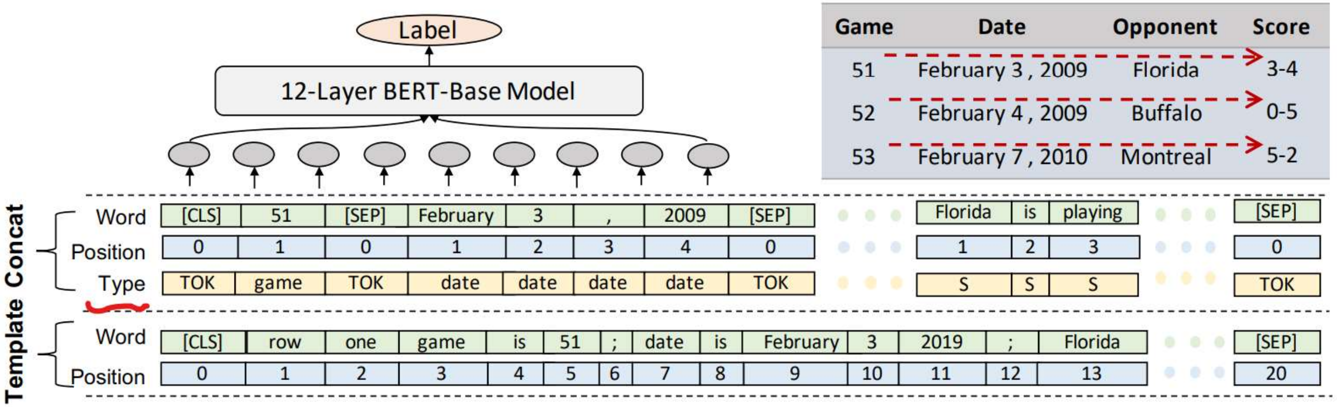
(1) Concatenation

(2) Template

2. concatenate $\tilde{\mathbf{T}}$ with $\mathbf{S}$

$$H = f_{BERT}([\tilde{\mathbf{T}}, S])$$

$$p_\theta(\tilde{\mathbf{T}}, S) = \sigma(f_{MLP}(H))$$



3. Classify as ENTAILED when $p_\theta > 0.5$

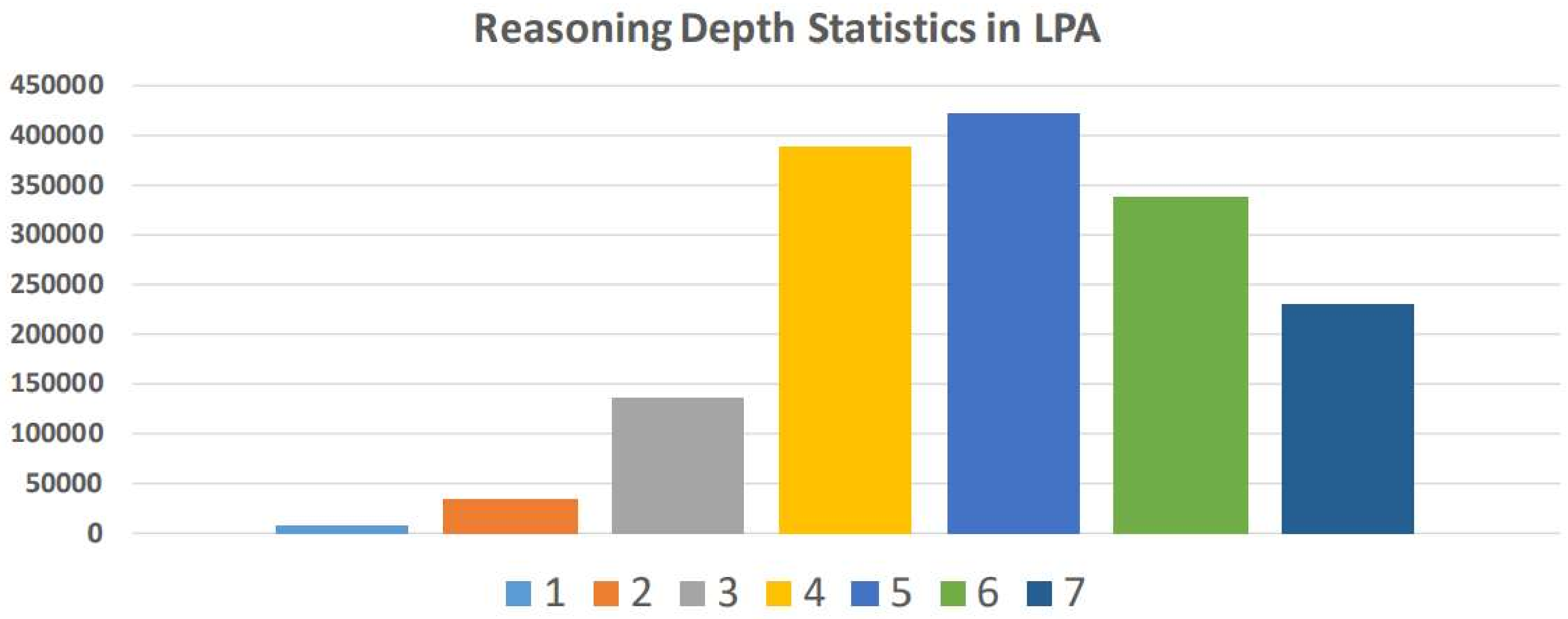
Experiments

Model	Val	Test	Test (simple)	Test (complex)	Small Test
BERT classifier w/o Table	50.9	50.5	51.0	50.1	50.4
Table-BERT-Horizontal-F+T-Concatenate	50.7	50.4	50.8	50.0	50.3
Table-BERT-Vertical-F+T-Template	56.7	56.2	59.8	55.0	56.2
Table-BERT-Vertical-T+F-Template	56.7	57.0	60.6	54.3	55.5
Table-BERT-Horizontal-F+T-Template	66.0	65.1	79.0	58.1	67.9
Table-BERT-Horizontal-T+F-Template	**66.1**	**65.1**	**79.1**	**58.2**	**68.1**
NSM w/ RL (Binary Reward)	54.1	54.1	55.4	53.1	55.8
NSM w/ LPA-guided ML + RL	63.2	63.5	77.4	56.1	66.9
LPA-Voting w/o Discriminator	57.7	58.2	68.5	53.2	61.5
LPA-Weighted-Voting	62.5	63.1	74.6	57.3	66.8
LPA-Ranking w/ Discriminator	**65.2**	65.0	78.4	**58.5**	68.6
LPA-Ranking w/ Discriminator (Caption)	65.1	**65.3**	**78.7**	**58.5**	**68.9**
Human Performance	-	-	-	-	**92.1**

Only 58% of sentences have been correctly linked
systematic search has a recall of 51%

Experiments

- **Reasoning depth**



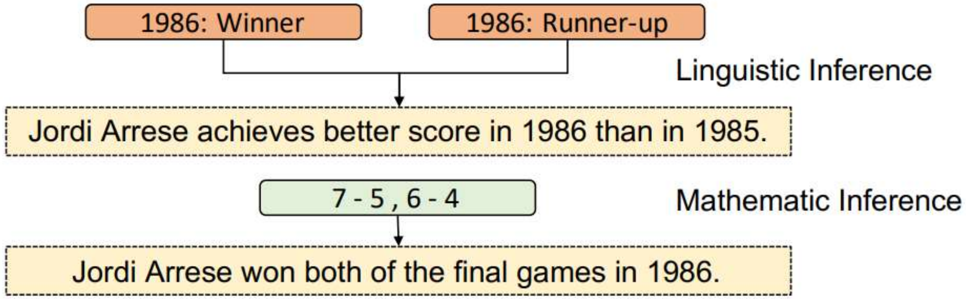
Experiments

- **Error analysis**

1. Symbolic

Jordi Arrese

outcome	date	tournament	surface	partner	opponents in the final	score in the final
runner - up	1985	Bologna , Italy	clay	Alberto Tous	Paolo Canè Simone Colombo	5 - 7 , 4 - 6
winner	1986	Bordeaux , France	clay	David De Miguel	Ronald Agénor Mansour Bahrami	7 - 5 , 6 - 4
winner	1989	Prague , Czechoslovakia	clay	Horst Skoff	Petr Korda Tomáš šmíd	6 - 4 , 6 - 4



Experiments

- **Error analysis**

2. BERT

Jordi Arrese

outcome	date	tournament	surface	partner	opponents in the final	score in the final
runner - up	1985	Bologna , Italy	clay	Alberto Tous	Paolo Canè Simone Colombo	5 - 7 , 4 - 6
winner	1986	Bordeaux , France	clay	David De Miguel	Ronald Agénor Mansour Bahrami	7 - 5 , 6 - 4
winner	1989	Prague , Czechoslovakia	clay	Horst Skoff	Petr Korda Tomáš šmíd	6 - 4 , 6 - 4

Template

Given the table titled “Jordi Arrese”, in row one, the outcome is runner-up, the date is 1985, … , the surface is clay …. …… , In row two, the outcome is … , the surface is clay. In row three, the outcome is …, … the surface is clay.

Long Dependency

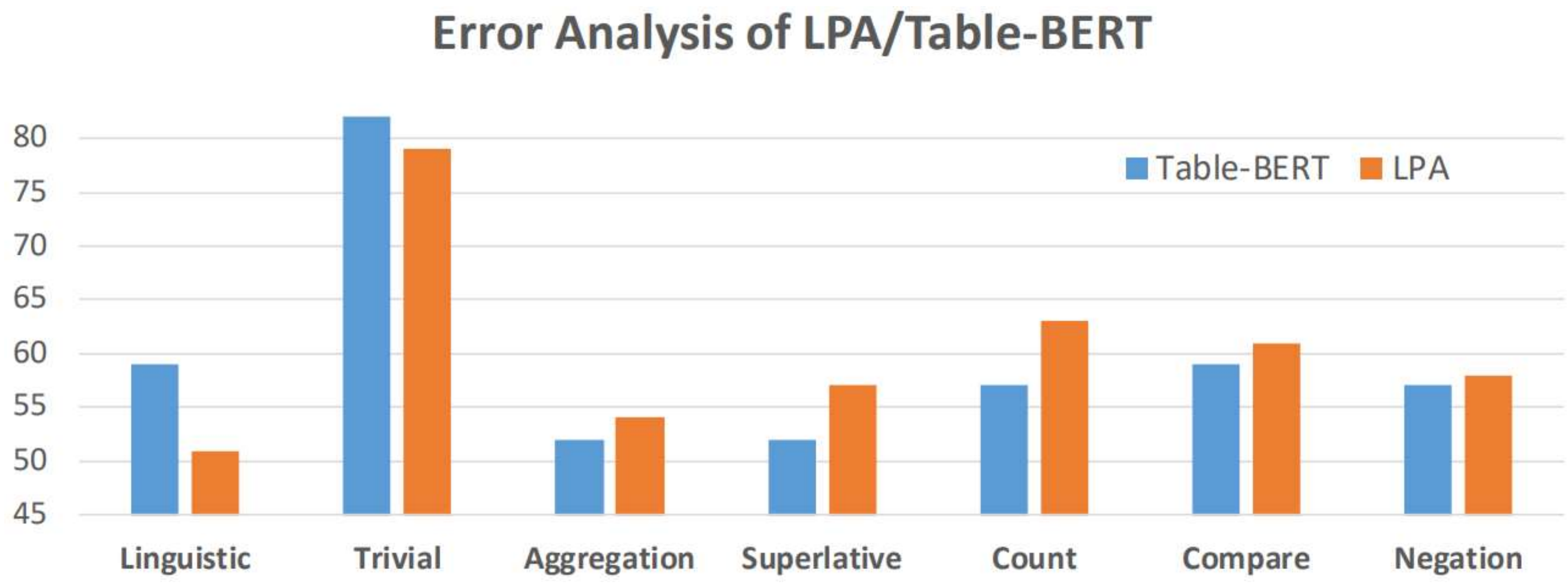
The three “Clay” are separated by more over 20 words

Jordi Arrese played all of his games on clay surface.

Experiments

- **Error analysis**

3. Statistics

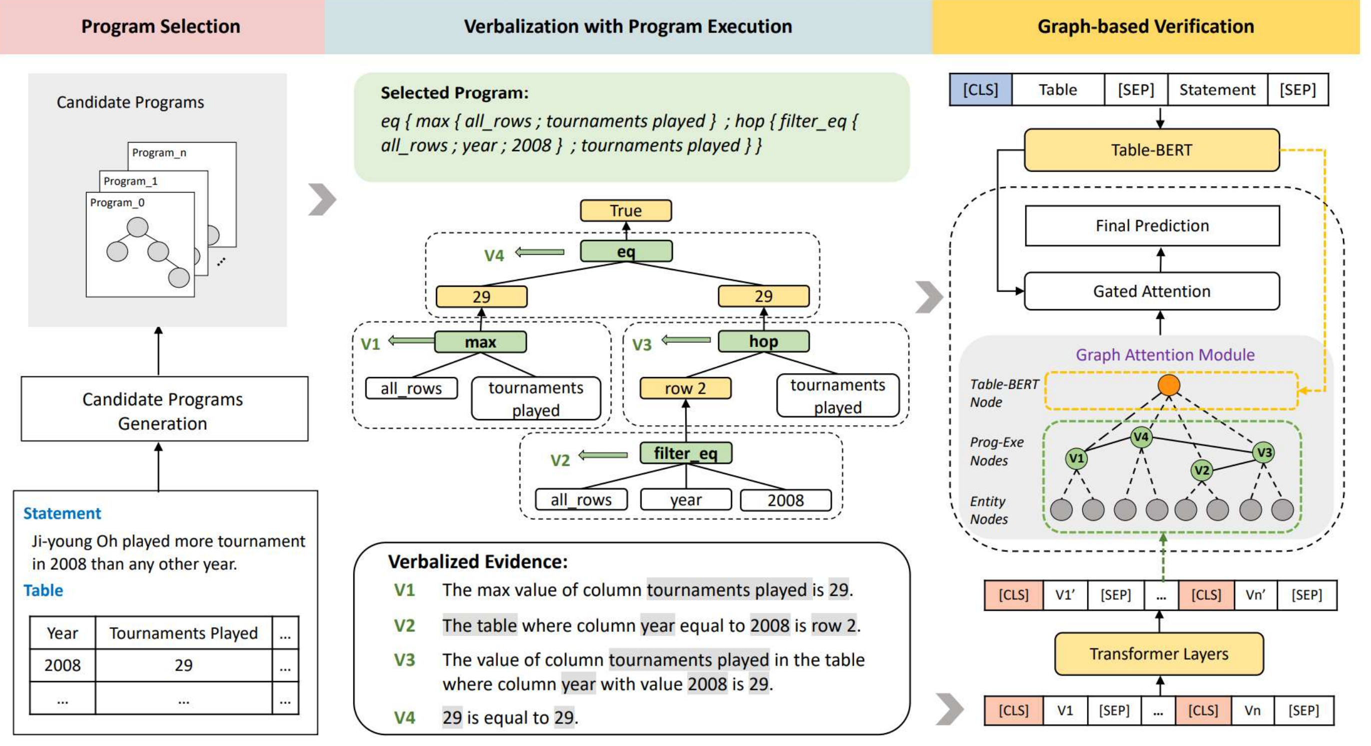


Program Enhanced Fact Verification with Verbalization and Graph Attention Network

Xiaoyu Yang[†,*] **Feng Nie**[§,*] **Yufei Feng**[†], **Quan Liu**[‡], **Zhigang Chen**[‡], **Xiaodan Zhu**[†]
[†] ECE & Ingenuity Labs Research Institute, Queen's University
[§] Sun Yat-sen University
[‡] State Key Laboratory of Cognitive Intelligence, iFLYTEK Research

Model

- **Prog**ram-enhanced **V**erbalization and **G**raph **AT**tention Network



Model

- **Program selection**

Select z^* from candidate programs Z

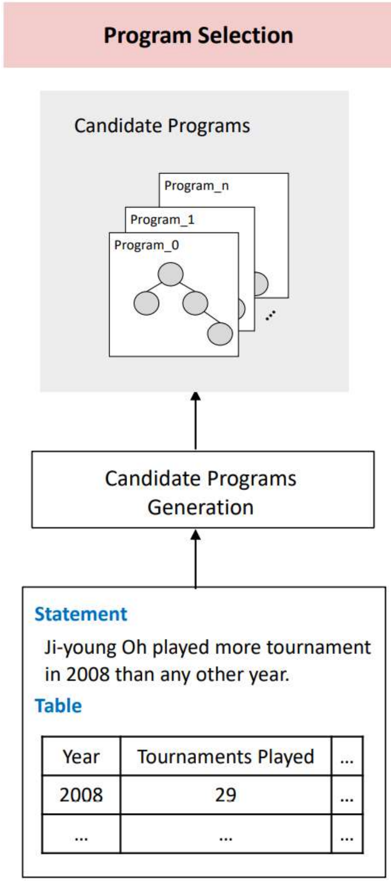
Former method:

1. Only one of the label-consistent program is correct

2. Consider every program in training but only one most relevant program selected in testing

$$p_\theta(z|S,T) = \sigma(W_r \boldsymbol{h})$$

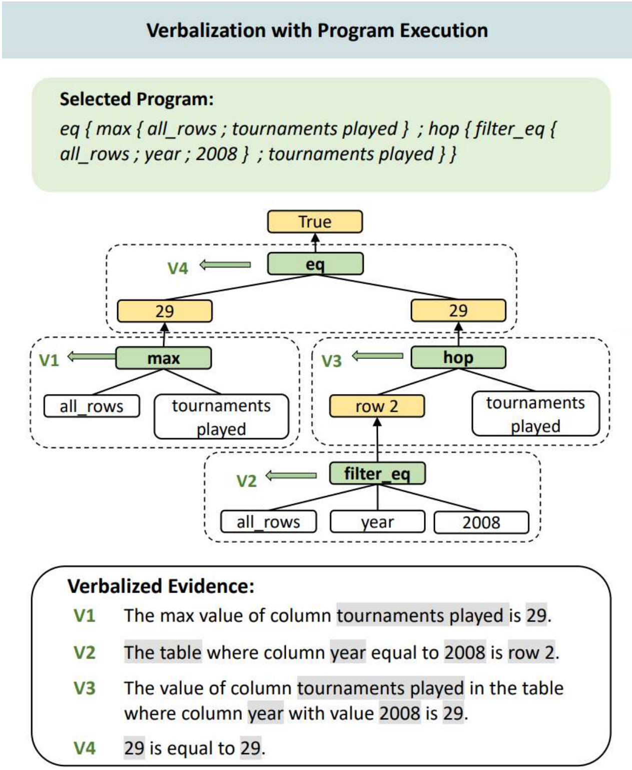
Margin loss $J(\theta) = \max\left(p_\theta(z'_{neg}|S,T) - p_\theta(z'_{pos}|S,T) + \gamma, 0\right)$



Model

- **Verbalization with program execution**

Convert the execution into natural language sentences



Algorithm 1 Verbalization

Require Statement and evidence table pair (S, T), and parsed program $z^* = \{op_i\}_{i=1}^{M}$; Pre-defined operator $P = \{p_i\}_{i=1}^{R}$; A template function $\mathcal{F}(.)$ maps operation and operation results into sentences.

```
function VERBALIZATION(op, ret)
    args = {}, verb_args = {}
    for a_j in arguments of operation op do
        if a_j is an operator in P then
            arg_ans, verb_arg = VERBALIZATION(a_j, ret)
            args ← args ∪ arg_ans
            verb_args ← verb_args ∪ verb_arg
        else
            args ← args ∪ a_j
            verb_args ← verb_args ∪ str(a_j)
        end if
    end for
    Apply operation (op.t, args) over evidence table T, obtain operation result ans
    Apply F(op.t, verb_args, ans), obtain verbalized operation result verb_ans and verbalized operation verb_op
    Update ret ← ret ∪ verb_ans
    Return ans, verb_op
end function
```

Set verbalized program execution $ret = \{\}$
VERBALIZATION(op_1, ret)
Return ret

Post-order traversal

Convert

Model

- **Graph-based Verification Network**

1.Definition

Nodes:

1) verbalized program executions $(n_0, \dots, n_{M-1})$

2) program entities $(n_M, \dots, n_{K-2})$

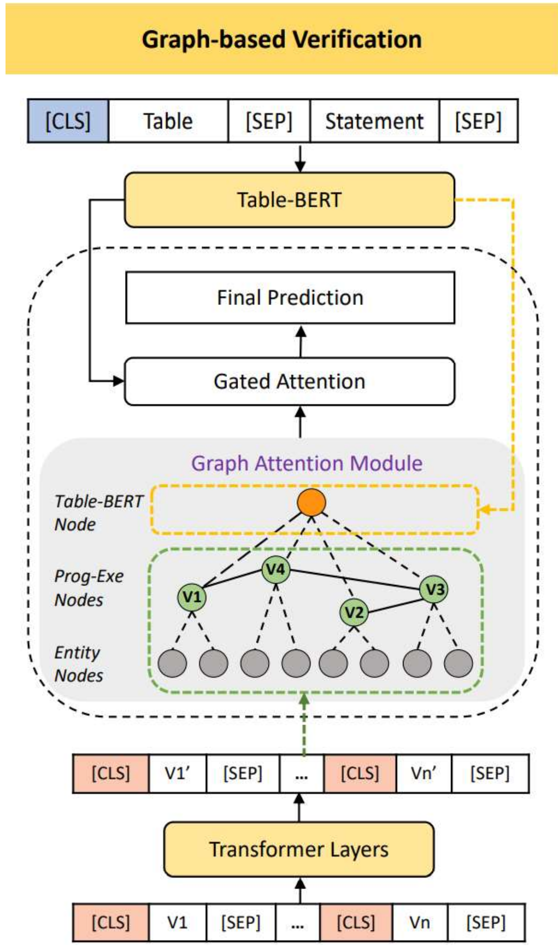
3) utilize information in table and statements n_{K-1}

Edges:

1) between executions

2) between execution and entity

3) between execution and Table-BERT node



Model

2.Graph construction and initialization

- Table-BERT node:

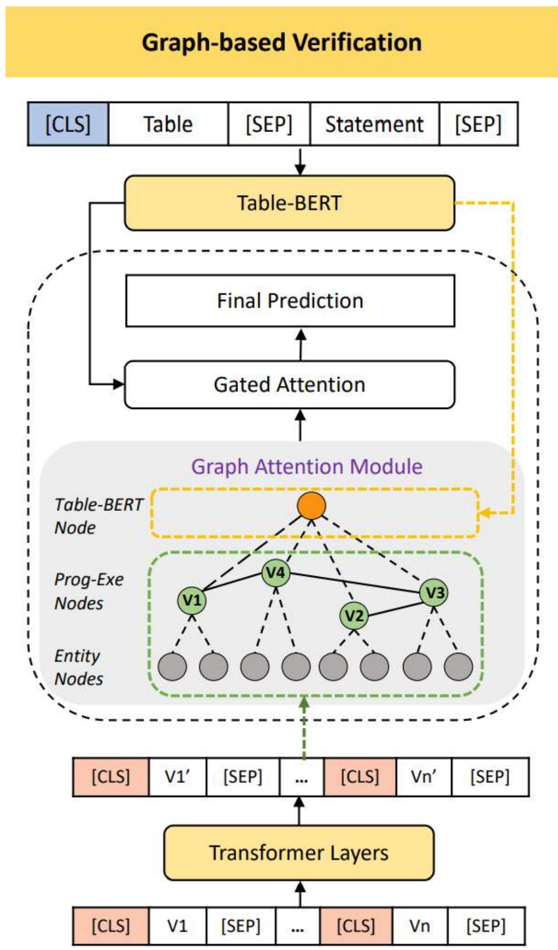
$$\boldsymbol{h}_{K-1} = f_{BERT}([\widetilde{\boldsymbol{T}}; S])$$

- Prog-Exec node: document-level BERT*

  [CLS] and [SEP] for every sentence

- Entity node:

  take the contextualized embeddings at positions corresponding to the entities in the top layer of BERT (average pooling for multiple words)



*Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. EMNLP-IJCNLP.

Model

3.Reasoning with graph attentions

Propagation:

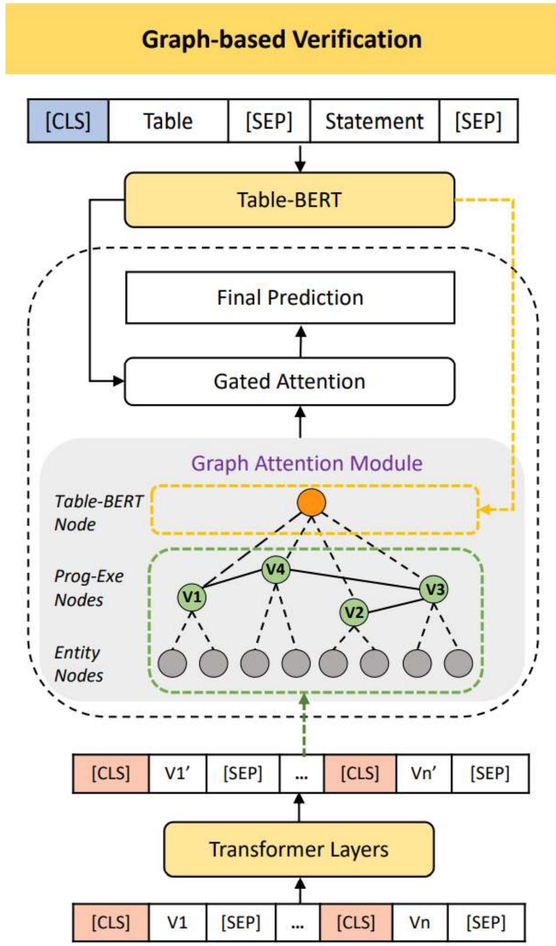
Edge: shared attention $e_{ij} = a(\boldsymbol{U}\boldsymbol{h}_i, \boldsymbol{U}\boldsymbol{h}_j)$

Normalized attention coefficient: $$\alpha_{ij}^d = \frac{exp(e_{ij})}{\sum_{k=1}^{K} A_{i,k}^d exp(e_{ik})}$$

Update node: $$\boldsymbol{h}_i^{new} = f(\Big\Vert_{d=1}^{D} \sigma(\sum_{j \in \mathcal{N}_i^d} \alpha_{ij}^d \boldsymbol{W}\boldsymbol{h}_j))$$

Gated attention: $$\boldsymbol{h}_{final} = \sum_{i=0}^{M-1} p_i \boldsymbol{h}_i{}^{new}; p_i = \sigma(\boldsymbol{h}_{K-1}^T \boldsymbol{h}_i{}^{new}),$$

$$y = \sigma(\boldsymbol{W_f}([\boldsymbol{h}_{final} \Vert \boldsymbol{h}_{K-1}]))$$



Experiment

- **Overall performance**

Model	Val	Test	Test (simple)	Test (complex)	Small Test
Human Performance	-	-	-	-	92.1
Table-BERT-Horizontal-S+T-Concatenate	50.7	50.4	50.8	50.0	50.3
Table-BERT-Vertical-S+T-Template	56.7	56.2	59.8	55.0	56.2
Table-BERT-Vertical-T+S-Template	56.7	57.0	60.6	54.3	55.5
Table-BERT-Horizontal-S+T-Template	66.0	65.1	79.0	58.1	67.9
Table-BERT-Horizontal-T+S-Template	66.1	65.1	79.1	58.2	68.1
LPA-Voting w/o Discriminator	57.7	58.2	68.5	53.2	61.5
LPA-Weighted-Voting	62.5	63.1	74.6	57.3	66.8
LPA-Ranking w/ Discriminator	65.2	65.0	78.4	58.5	68.6
LogicalFactChecker (Zhong et al., 2020)	71.8	71.7	85.4	65.1	74.3
ProgVGAT	**74.9**	**74.4**	**88.3**	**67.6**	**76.2**

- **Effect of program operations**

Model	Val	Test
Table-BERT w/ prog	70.3	70.0
LogicalFactChecker	71.8	71.7
Table-BERT w/ verb. prog	71.8	71.6
Table-BERT w/ verb. prog exec	72.4	72.2
ProgVGAT	**74.9**	**74.4**

Experiment

- **Effect of graph attention**

Model	Val	Test
ProgVGAT w/o graph attention	73.6	73.4
ProgVGAT	**74.9**	**74.4**

- **Effect of derived programs**

<table>
<tr><th rowspan="2" colspan="3"></th><th colspan="3">Final Verification</th></tr>
<tr><th>Val</th><th>Test</th><th>ΔTest</th></tr>
<tr><td rowspan="2">LPA
w/ CE</td><td>Val</td><td>Test</td><td rowspan="2">73.3</td><td rowspan="2">72.8</td><td rowspan="2">-</td></tr>
<tr><td>65.2</td><td>65.0</td></tr>
<tr><td rowspan="2">LPA+ BERT
w/ CE</td><td>Val</td><td>Test</td><td rowspan="2">73.9</td><td rowspan="2">73.4</td><td rowspan="2">+0.6</td></tr>
<tr><td>67.7</td><td>67.3</td></tr>
<tr><td rowspan="2">LPA +BERT
w/ Margin loss</td><td>Val</td><td>Test</td><td rowspan="2">74.9</td><td rowspan="2">74.4</td><td rowspan="2">+1.6</td></tr>
<tr><td>69.4</td><td>68.5</td></tr>
</table>

TAPAS: Weakly Supervised Table Parsing via Pre-training

Jonathan Herzig[1,2], Paweł Krzysztof Nowak[1], Thomas Müller[1],
Francesco Piccinno[1], Julian Martin Eisenschlos[1]

[1]Google Research
[2]School of Computer Science, Tel-Aviv University

Understanding tables with intermediate pre-training

Julian Martin Eisenschlos, Syrine Krichene, Thomas Müller

Google Research, Zürich

Model

Flatten the table into a sequence of words, split words into word pieces (tokens) and concatenate the question tokens before the table tokens

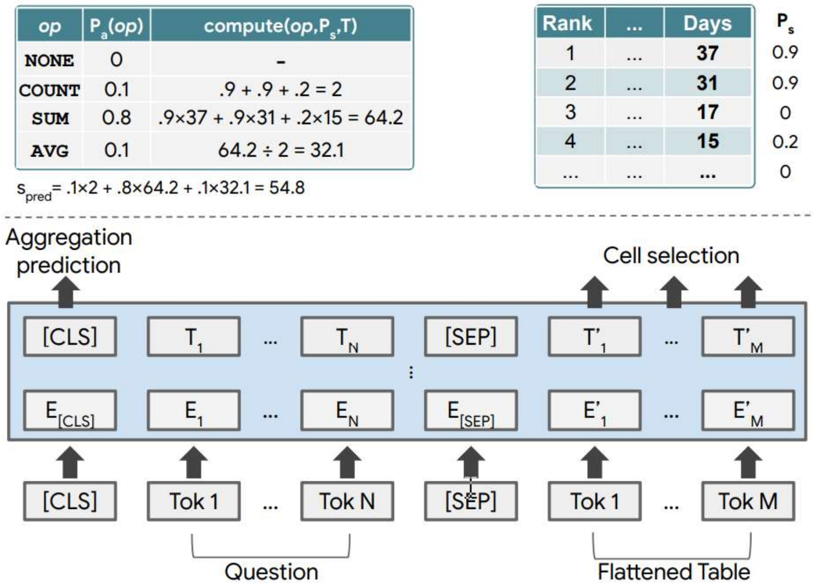
For training set $\{(x_i, T_i, y_i)\}_{i=1}^{N}$: utterance x_i, table T_i, denotation y_i

Translate y to a tuple (C, s) : cell coordinates C (and a scalar s when y is a scalar)

1. Additional embeddings

2. Cell selection

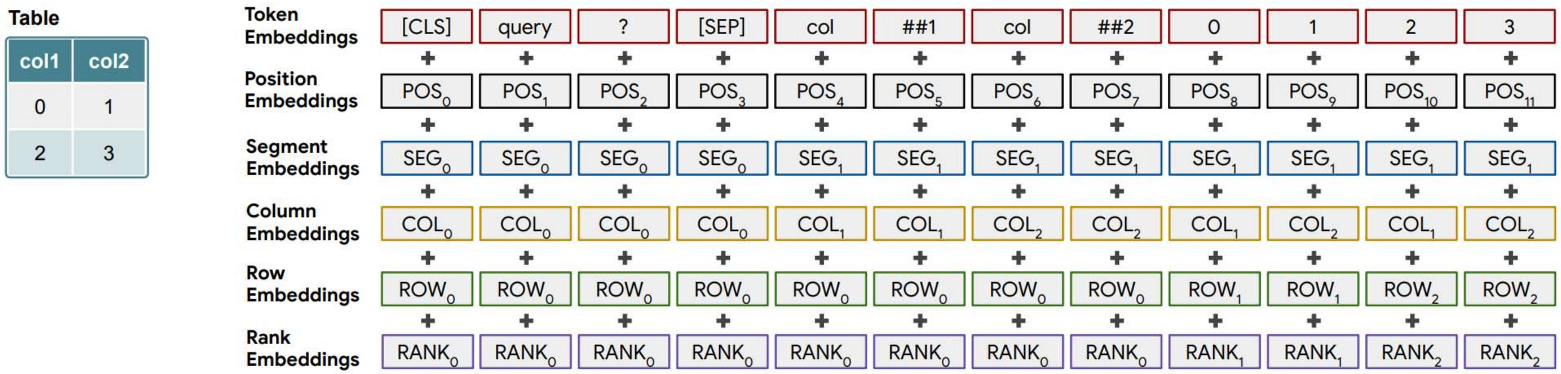
3. Aggregation operator prediction



Model

1. Additional embeddings

- **Position ID:** same as in BERT
- **Segment ID:** 0 for the question, 1 for the table header and cells
- **Column/Row ID:** index of the column/row, 0 for question
- **Rank ID:** if column values can be floats or dates, 0 for not comparable, 1 for smallest, $i + 1$ for rank i



Model

2. Cell selection

1) token logit: BERT output into one linear layer

2) cell logit: average tokens logits in the cell, one linear layer $p_S^{(c)}$

3) column logit: average cell logits in the column, one linear layer & softmax $p_{col}^{(co)}$

(one additional logit for selecting no column/cell)

Select the column with most cells in C

- **Loss function:**

1) column: $$\mathcal{J}_{\text{columns}} = \frac{1}{|\text{Cols}|} \sum_{\text{co} \in \text{Cols}} \text{CE}(p_{\text{col}}^{(\text{co})}, \mathbb{1}_{\text{co}=\text{col}})$$

2) cell: $$\mathcal{J}_{\text{cells}} = \frac{1}{|\text{Cells(col)}|} \sum_{c \in \text{Cells(col)}} \text{CE}(p_{\text{s}}^{(c)}, \mathbb{1}_{c \in C})$$

$$\mathcal{J}_{\text{CS}} = \mathcal{J}_{\text{columns}} + \mathcal{J}_{\text{cells}} + \alpha \mathcal{J}_{\text{aggr}}$$

3) aggregation (no operation occurs, use op_0): $$\mathcal{J}_{\text{aggr}} = -\log p_{\text{a}}(op_0)$$

Model

3. Aggregation operator prediction

BERT output of [CLS] token into a linear layer & softmax $p_a(op)$

Applying aggregation over cells $p_S^{(c)} > 0.5$

op	$\text{compute}(op, p_\text{s}, T)$
COUNT	$\sum_{c \in T} p_\text{s}^{(c)}$
SUM	$\sum_{c \in T} p_\text{s}^{(c)} \cdot T[c]$
AVERAGE	$\frac{\text{compute}(\texttt{SUM}, p_\text{s}, T)}{\text{compute}(\texttt{COUNT}, p_\text{s}, T)}$

- **Scalar answer**

normalized probability excluding NONE $\hat{p}_\text{a}(op_i) = \frac{p_\text{a}(op_i)}{\sum_{i=1} p_\text{a}(op_i)}$

predict result $s_\text{pred} = \sum_{i=1} \hat{p}_\text{a}(op_i) \cdot \text{ compute } (op_i, p_\text{s}, T)$

scalar answer loss $a = |s_\text{pred} - s|$ $\mathcal{J}_\text{scalar} = \begin{cases} 0.5 \cdot a^2 & a \leq \delta \\ \delta \cdot a - 0.5 \cdot \delta^2 & \text{otherwise} \end{cases}$

aggregation loss* $\mathcal{J}_\text{aggr} = -\log\left(\sum_{i=1} p_\text{a}(op_i)\right)$ $\mathcal{J}_\text{SA} = \mathcal{J}_\text{aggr} + \beta \mathcal{J}_\text{scalar}$

Model

- **Ambiguous answer**

Scalar answer can be selected or inferenced through aggregation

Dynamically let the model choose the supervision according to policy

Use cell selection if $p_a(op_0) \geq S$, or scalar answer otherwise

Table

Rank	Name	No. of reigns	Combined days
1	Lou Thesz	3	3,749
2	Ric Flair	8	3,103
3	Harley Race	7	1,799
4	Dory Funk Jr.	1	1,563
5	Dan Severn	2	1,559
6	Gene Kiniski	1	1,131

Example questions

#	Question	Answer	Example Type
1	*Which wrestler had the most number of reigns?*	Ric Flair	Cell selection
2	*Average time as champion for top 2 wrestlers?*	AVG(3749,3103)=3426	Scalar answer
3	*How many world champions are there with only one reign?*	COUNT(Dory Funk Jr., Gene Kiniski)=2	Ambiguous answer
4	*What is the number of reigns for* Harley Race?	7	Ambiguous answer
5	*Which of the following wrestlers were ranked in the bottom 3?*	{Dory Funk Jr., Dan Severn, Gene Kiniski}	Cell selection
	Out of these, who had more than one reign?	Dan Severn	Cell selection

Methods

- **Pre-training tasks**

Learn correlations between text and table, and between cells of a columns and their header

Extract 6.2M tables and 21.3M snippets from relevant text

Mask-LM: Whole word masking for the text, whole cell masking to the tables

1) Counterfactual statements
2) Synthetic statements

Methods

1) Counterfactual statements

Create a minimally differing refuted example from positive examples

- Replace mention

occur in the same column

- Supporting mention

entity that occurs in the same row

e.g.

[Greg Norman] is [Australian]

Rank	Player	Country	Earnings	Events	Wins
1	Greg Norman	Australia	1,654,959	16	3
2	Billy Mayfair	United States	1,543,192	28	2
3	Lee Janzen	United States	1,378,966	28	3
4	Corey Pavin	United States	1,340,079	22	2
5	Steve Elkington	Australia	1,254,352	21	2

Methods

2) Synthetic statements

Improve the handling of numerical operations and comparisons

⟨statement⟩	→	⟨expr⟩⟨compare⟩⟨expr⟩
⟨expr⟩	→	⟨select⟩ when ⟨where⟩ | ⟨select⟩
⟨select⟩	→	⟨column⟩ | the ⟨aggr⟩ of ⟨column⟩ | the count
⟨where⟩	→	⟨column⟩⟨compare⟩⟨value⟩ | ⟨where⟩ and ⟨where⟩
⟨aggr⟩	→	first | last | lowest | greatest | sum | average | range
⟨compare⟩	→	is | is greater than | is less than
⟨value⟩	→	⟨string⟩ | ⟨number⟩

Rank	Player	Country	Earnings	Events	Wins
1	Greg Norman	Australia	1,654,959	16	3
2	Billy Mayfair	United States	1,543,192	28	2
3	Lee Janzen	United States	1,378,966	28	3
4	Corey Pavin	United States	1,340,079	22	2
5	Steve Elkington	Australia	1,254,352	21	2

Synthetic: 2 is less than wins when Player is Lee Janzen.

The sum of Earnings when Country is Australia is $2,909,311$.

Methods

- **Table pruning**

1) Selecting the first token of every cell, then the second until reach the maximal length

2) Ranking columns by relevance score and added in order of decreasing relevance

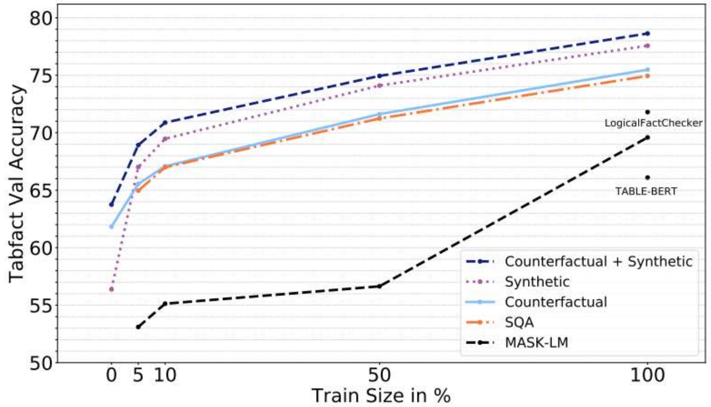
Jaccard coefficient: $\left|\frac{T_S \cap T_C}{T_S \cup T_C}\right|$

Result

- **TABFACT**

Model	Val	Test	Test simple	Test complex	Test small
BERT classifier w/o Table	50.9	50.5	51.0	50.1	50.4
TABLE-BERT-Horizontal-T+F-Template	66.1	65.1	79.1	58.2	68.1
LPA-Ranking w/ Discriminator (Caption)	65.1	65.3	78.7	58.5	68.9
LFC (program from LPA)	71.7	71.6	85.5	64.8	74.2
LFC (program from Seq2Action)	71.8	71.7	85.4	65.1	74.3
ProgVGAT	74.9	74.4	88.3	67.6	76.2
OURS-Base-MASK-LM	$69.6_{\pm 4.4}$	$69.9_{\pm 3.8}$	$82.0_{\pm 5.9}$	$63.9_{\pm 2.8}$	$72.2_{\pm 4.7}$
OURS-Base-SQA	$74.9_{\pm 0.2}$	$74.6_{\pm 0.2}$	$87.2_{\pm 0.2}$	$68.4_{\pm 0.4}$	$77.3_{\pm 0.3}$
OURS-Base-Counterfactual	$75.5_{\pm 0.5}$	$75.2_{\pm 0.4}$	$87.8_{\pm 0.4}$	$68.9_{\pm 0.5}$	$77.4_{\pm 0.3}$
OURS-Base-Synthetic	$77.6_{\pm 0.2}$	$77.9_{\pm 0.3}$	$89.7_{\pm 0.4}$	$72.0_{\pm 0.2}$	$80.4_{\pm 0.2}$
OURS-Base-Counterfactual + Synthetic	$\mathbf{78.6}_{\pm 0.3}$	$\mathbf{78.5}_{\pm 0.3}$	$\mathbf{90.5}_{\pm 0.4}$	$\mathbf{72.5}_{\pm 0.3}$	$\mathbf{81.0}_{\pm 0.3}$
OURS-Large-Counterfactual + Synthetic	$\mathbf{81.0}_{\pm 0.1}$	$\mathbf{81.0}_{\pm 0.1}$	$\mathbf{92.3}_{\pm 0.3}$	$\mathbf{75.6}_{\pm 0.1}$	$\mathbf{83.9}_{\pm 0.3}$
Human Performance	–	–	–	–	92.1

- **Zero-shot accuracy and low resource regimes**



Result

- **Ablations**

	SQA (SEQ)		WikiSQL		WikiTQ	
all	39.0		84.7		29.0	
-pos	36.7	-2.3	82.9	-1.8	25.3	-3.7
-ranks	34.4	-4.6	84.1	-0.6	30.7	+1.8
-{cols,rows}	19.6	-19.4	74.1	-10.6	17.3	-11.6
-table pre-training	26.5	-12.5	80.8	-3.9	17.9	-11.1
-aggregation	-		82.6	-2.1	23.1	-5.9

- **Limitations**

1) This model would fail to capture large tables or multiple tables

2) Its expressivity is limited to a form of AN aggregation over a subset of table cells

Thanks!