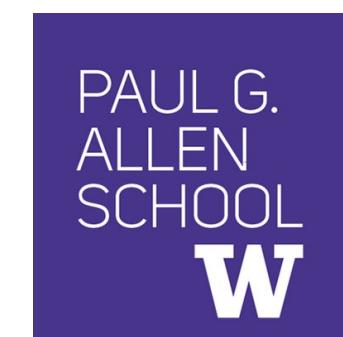


Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing

Paul G. Allen School of Computer Science and Engineering

Ph.D. Qualification Exam - Winter'21



Victoria Lin

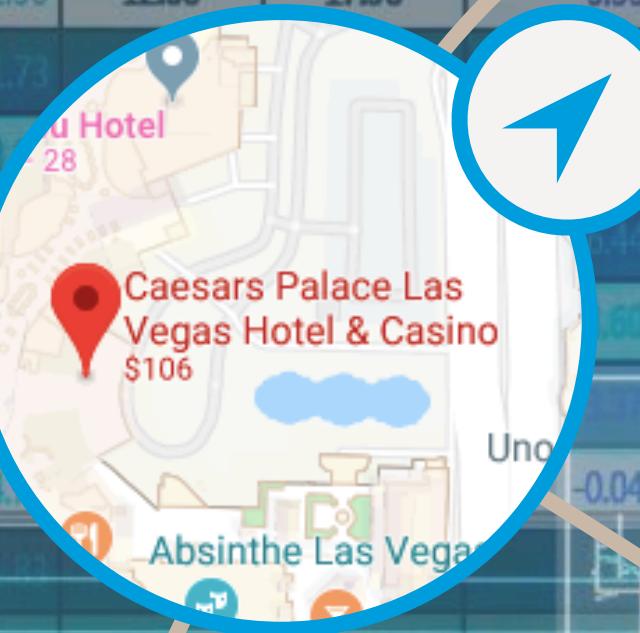


Work done at Salesforce AI Research

Personal



Intelligent



Conversational



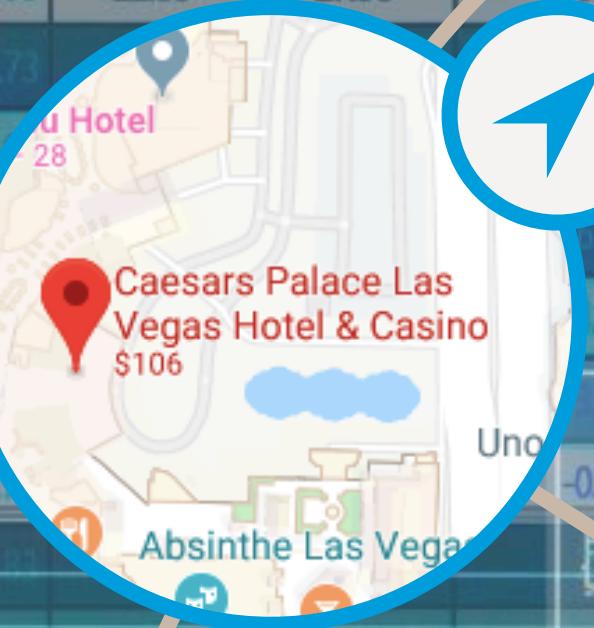
Tables and databases are commonly used data structures that power a variety of downstream applications.



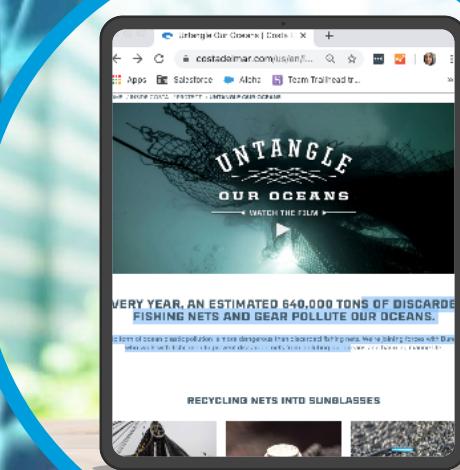
Personal



Intelligent



Conversational



The Internet contains billions of public tables. And more tabular data can be found on corporate intranets and personal devices.

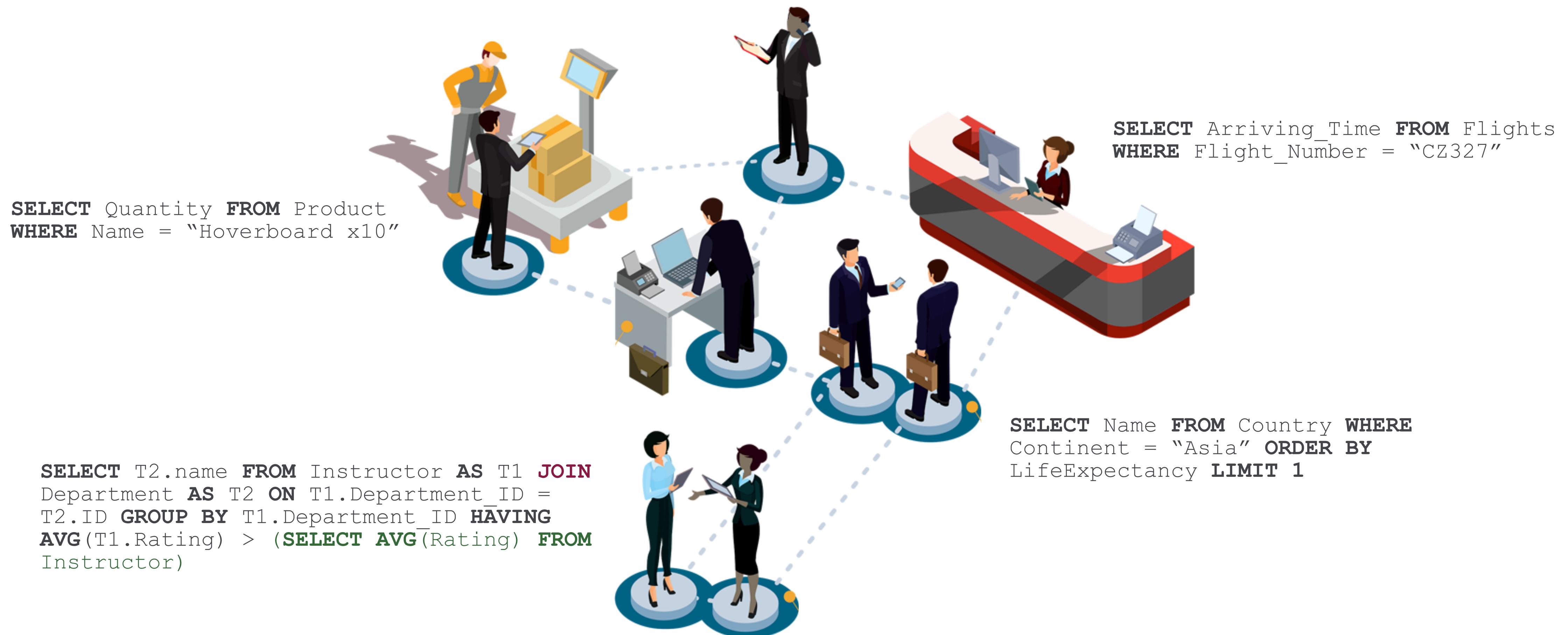
Natural Language Interface to Databases

Traditionally, users access databases using structured query language (SQL).



Natural Language Interface to Databases

Traditionally, users access databases using structured query language (SQL).



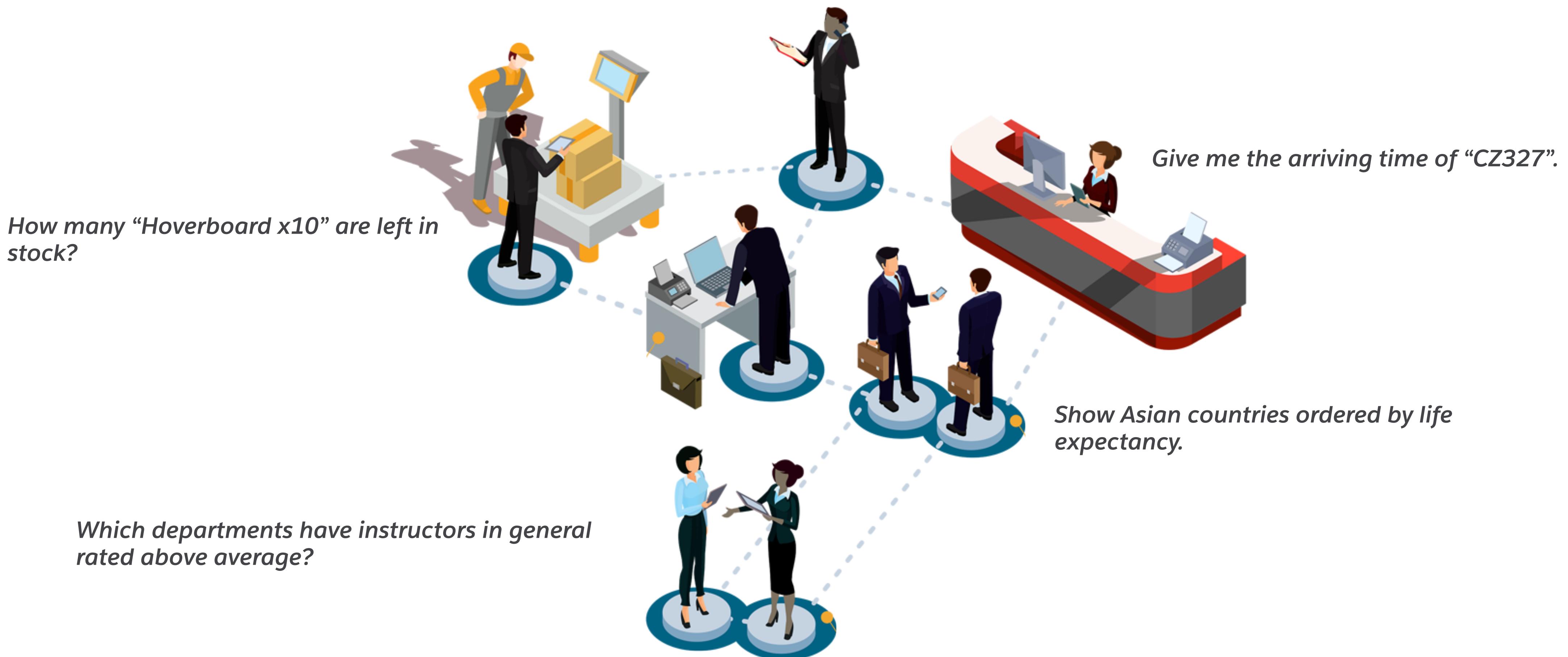
Natural Language Interface to Databases

Our goal is to learn semantic parsers that map natural language utterances to **executable** SQL queries for **any table/database**.



Natural Language Interface to Databases

Our goal is to learn semantic parsers that map natural language utterances to **executable SQL queries for any table/database**.





research

https://naturalsql.com

Photon About

real_estate_property

Tables

Other_Available_Features

feature_id	feature_type_code	feature_name	feature_description
2	Amenity	AirCon	Air Conditioning.
3	Amenity	Pool	Swimming Pool.
4	Security	BurglarAlarm	Burglar Alarm

Ref_Feature_Types

feature_type_code	feature_type_name
Amenity	Amenity, eg Pool.
Security	Security, eg Burglar Alarm.

Other_Property_Features

property_id	feature_id	property_feature_description
15	3	dolorem
12	4	earum
6	2	illo
7	2	hic
6	3	et

Properties

property_id	property_type_code	date_on_market	date_sold	property_name	property_address
1	House	1991-06-21 23:52:10	1979-05-13 16:58:06	park	4745 Emerson Stravenue Suite 829 South
2	House	1990-05-25 23:01:51	1990-11-14 19:16:38	the cole	098 Tremaine Highway Suite 569 South W
3	Other	1986-11-26 04:12:18	1981-06-26 21:28:28	prism	062 Micaela Court Apt. 707 Margretville, V
4	Field	2017-09-14 15:49:23	2003-02-27 18:17:11	riverside	49578 Ayden Mountains New Russellhave
5	Apartment	2016-05-06 16:53:39	2012-08-19 07:36:57	parc east	2765 Schulist Stream Lindmouth, UT 0339

Chat started by Photon • 3:43:58 AM

list name of all technicians and the machines they repair

```
SELECT technician.Name FROM repair_assignment JOIN machine ON repair_assignment.Machine_ID = machine.Machine_ID JOIN technician ON repair_assignment.technician_id = technician.technician_id
```

Did I get it right?

no

Sorry...

Please rephrase your previous question or ask a new question

List names of technician and the machines series they repaired

```
SELECT technician.Name, machine.Machine_series FROM repair_assignment JOIN machine ON repair_assignment.Machine_ID = machine.Machine_ID JOIN technician ON repair_assignment.technician_id = technician.technician_id
```

Did I get it right?

yes

Great!

Please input your question in NL or SQL to query the DB

Type Here

Text-to-SQL translation

Query arbitrary subsets of tables in any database

Support table join and other complex SQL operators



research

https://naturalsql.com

Photon About

real_estate_property

Tables

Other_Available_Features

feature_id	feature_type_code	feature_name	feature_description
2	Amenity	AirCon	Air Conditioning.
3	Amenity	Pool	Swimming Pool.
4	Security	BurglarAlarm	Burglar Alarm

Ref_Feature_Types

feature_type_code	feature_type_name
Amenity	Amenity, eg Pool.
Security	Security, eg Burglar Alarm.

Other_Property_Features

property_id	feature_id	property_feature_description
15	3	dolorem
12	4	earum
6	2	illo
7	2	hic
6	3	et

Properties

property_id	property_type_code	date_on_market	date_sold	property_name	property_address
1	House	1991-06-21 23:52:10	1979-05-13 16:58:06	park	4745 Emerson Stravenue Suite 829 South
2	House	1990-05-25 23:01:51	1990-11-14 19:16:38	the cole	098 Tremaine Highway Suite 569 South W
3	Other	1986-11-26 04:12:18	1981-06-26 21:28:28	prism	062 Micaela Court Apt. 707 Margretville, V
4	Field	2017-09-14 15:49:23	2003-02-27 18:17:11	riverside	49578 Ayden Mountains New Russellhave
5	Apartment	2016-05-06 16:53:39	2012-08-19 07:36:57	parc east	2765 Schulist Stream Lindmouth, UT 0339

Chat started by Photon • 3:43:58 AM

list name of all technicians and the machines they repair

```
SELECT technician.Name FROM repair_assignment JOIN machine ON repair_assignment.Machine_ID = machine.Machine_ID JOIN technician ON repair_assignment.technician_id = technician.technician_id
```

Did I get it right?

no

Sorry...

Please rephrase your previous question or ask a new question

List names of technician and the machines series they repaired

```
SELECT technician.Name, machine.Machine_series FROM repair_assignment JOIN machine ON repair_assignment.Machine_ID = machine.Machine_ID JOIN technician ON repair_assignment.technician_id = technician.technician_id
```

Did I get it right?

yes

Great!

Please input your question in NL or SQL to query the DB

Type Here

Text-to-SQL translation

Query arbitrary subsets of tables in any database

Support table join and other complex SQL operators

Outline

1. Problem Definition and Related Work

2. Approach

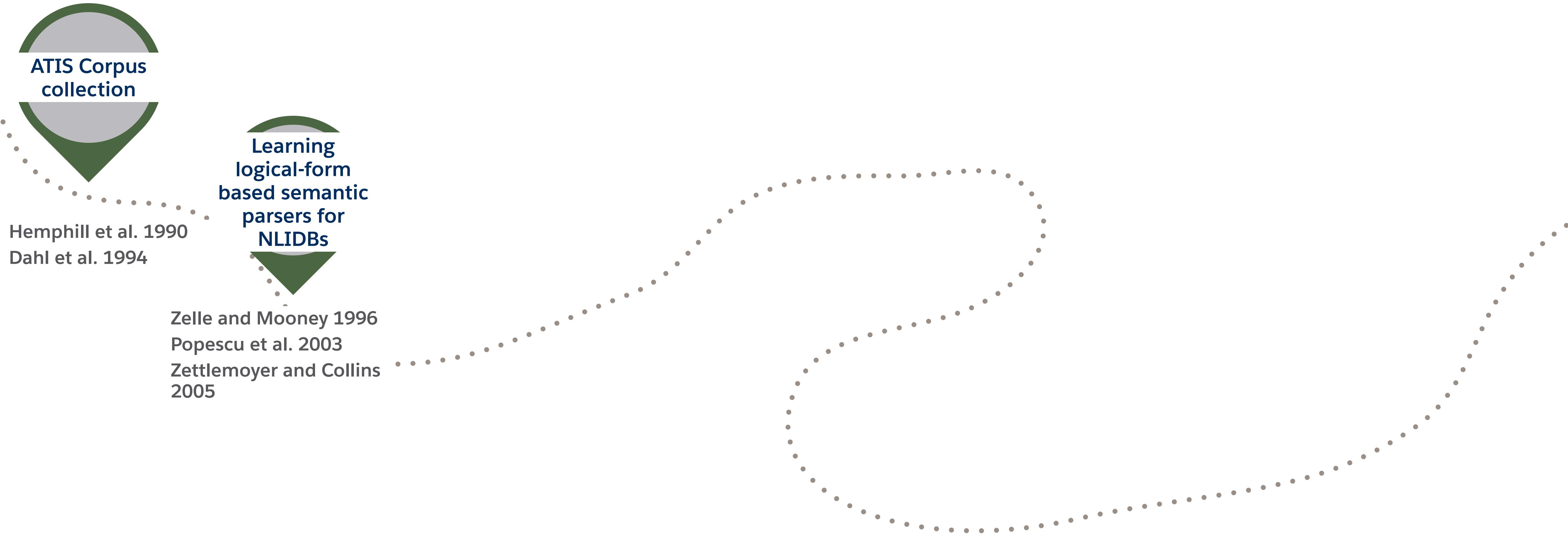
- BRIDGE encoder for jointly encoding text and relational databases
- BRIDGE decoder
 - A. Schema-consistency guided decoding
 - B. Static SQL correctness check

3. Experiments

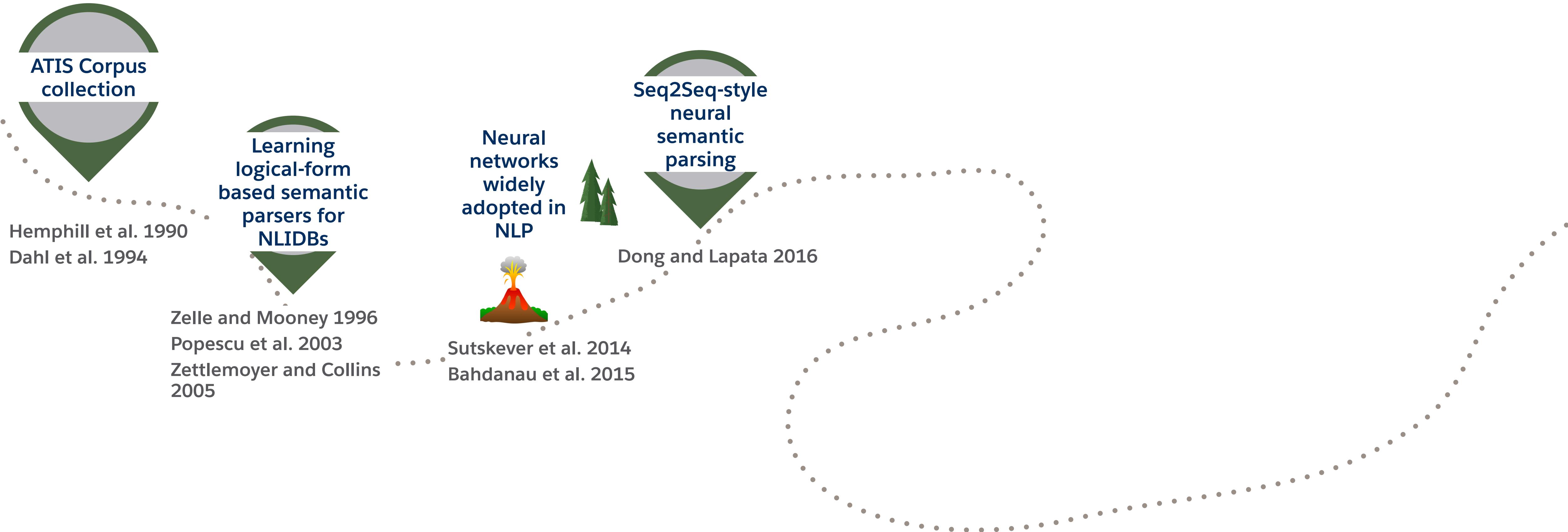
1. Ablation Study
2. Leaderboard Performances
3. Error Analysis

4. Conclusion

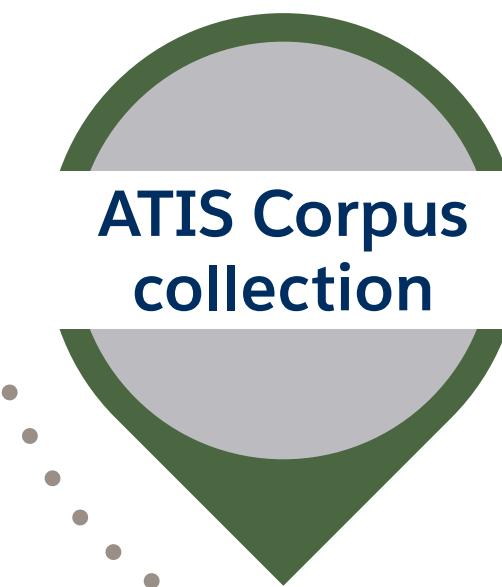
Literature Review



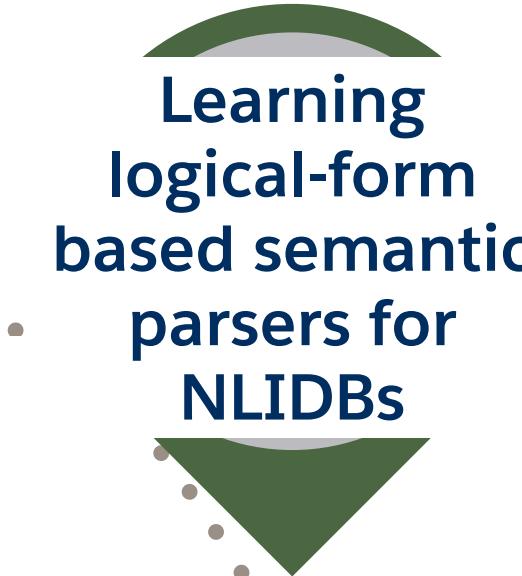
Literature Review



Literature Review

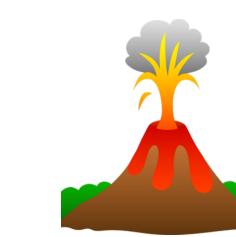


Hemphill et al. 1990
Dahl et al. 1994



Zelle and Mooney 1996
Popescu et al. 2003
Zettlemoyer and Collins 2005

Neural networks widely adopted in NLP

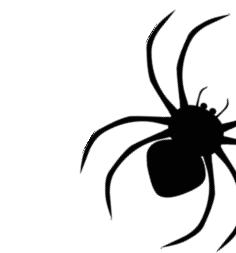


Sutskever et al. 2014
Bahdanau et al. 2015

Seq2Seq-style neural semantic parsing



Dong and Lapata 2016



Yu et al. 2018b

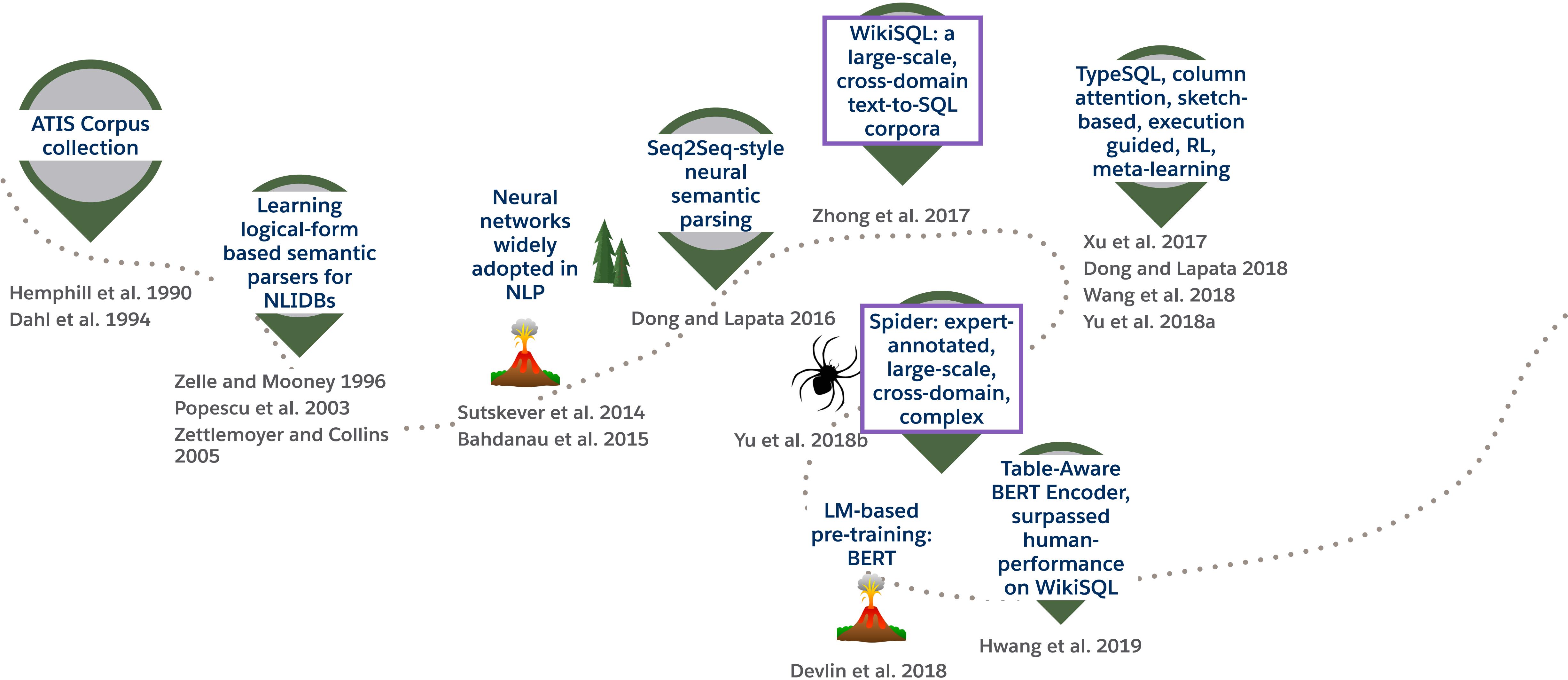
WikiSQL: a large-scale, cross-domain text-to-SQL corpora

Zhong et al. 2017

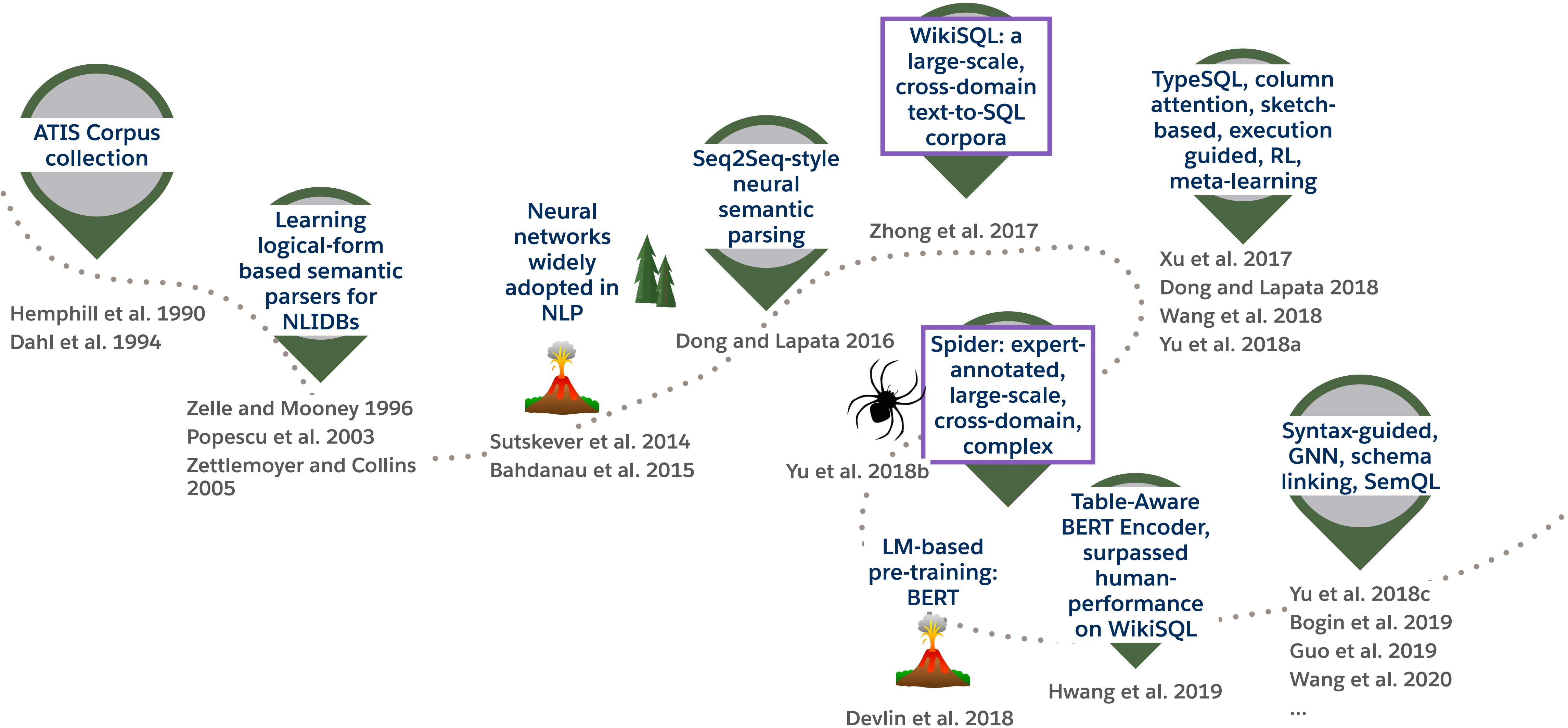
TypeSQL, column attention, sketch-based, execution guided, RL, meta-learning

Xu et al. 2017
Dong and Lapata 2018
Wang et al. 2018
Yu et al. 2018a

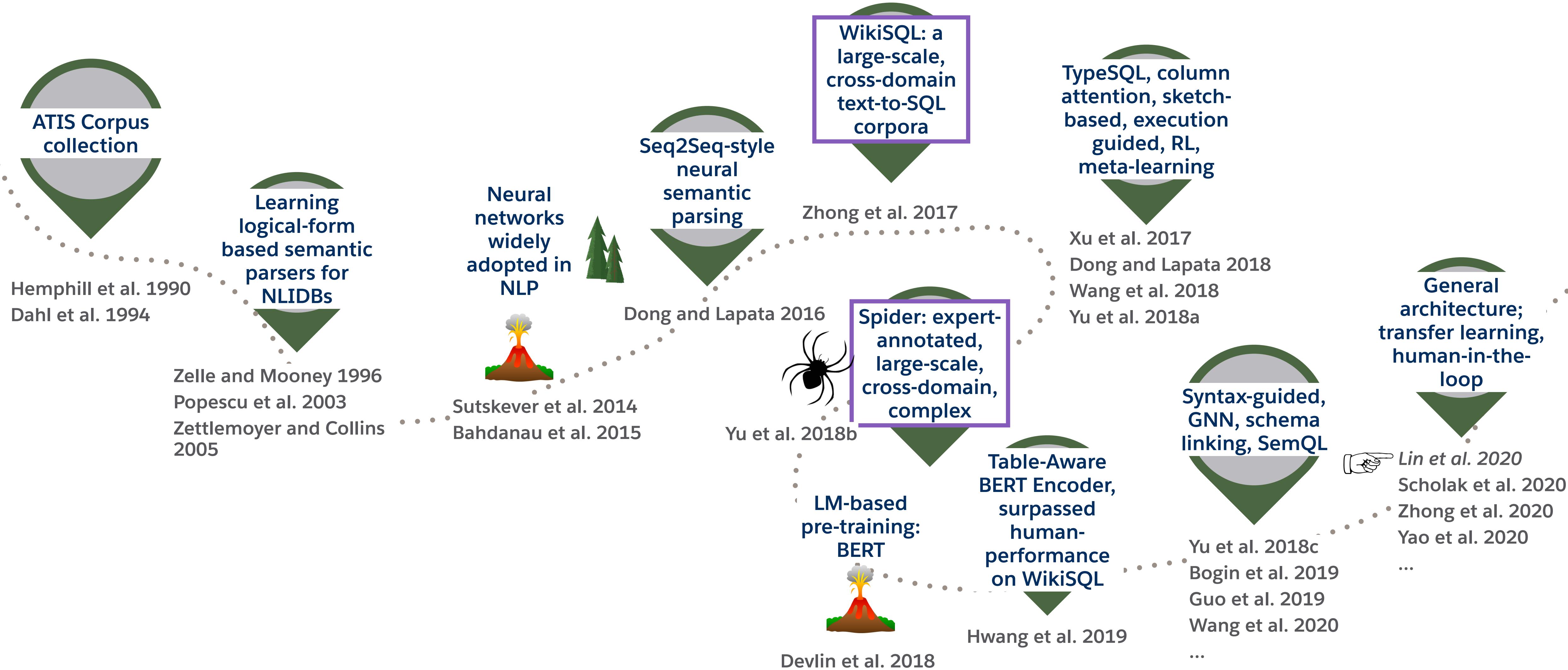
Literature Review



Literature Review



Literature Review



Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...



A table typically represents a type of entities (or events).

Relational Database Terms

User Profile ← Table Name

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

Table Header

Relational Database Terms

User Profile						
	Name	Nationality	PartitionID	# Followers		
UserID	Name	Nationality	PartitionID	# Followers		
103041	Smith, John	Canada	P10	3	...	
103042	Hanks, Tom	United States	P11	16.6M	...	
103043	Obama, Barack	United States	P11	127M	...	
...	

 A column name defines the semantic relationship between columns and between a column and the entity/event type represented by the table.

Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

Row



A row is typically an instantiation of the entity/event.

Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

column/field



Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

Integer
↑

Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

String
↑

Relational Database Terms

User Profile

UserID	Name	Nationality	PartitionID	# Followers	...
103041	Smith, John	Canada	P10	3	...
103042	Hanks, Tom	United States	P11	16.6M	...
103043	Obama, Barack	United States	P11	127M	...
...

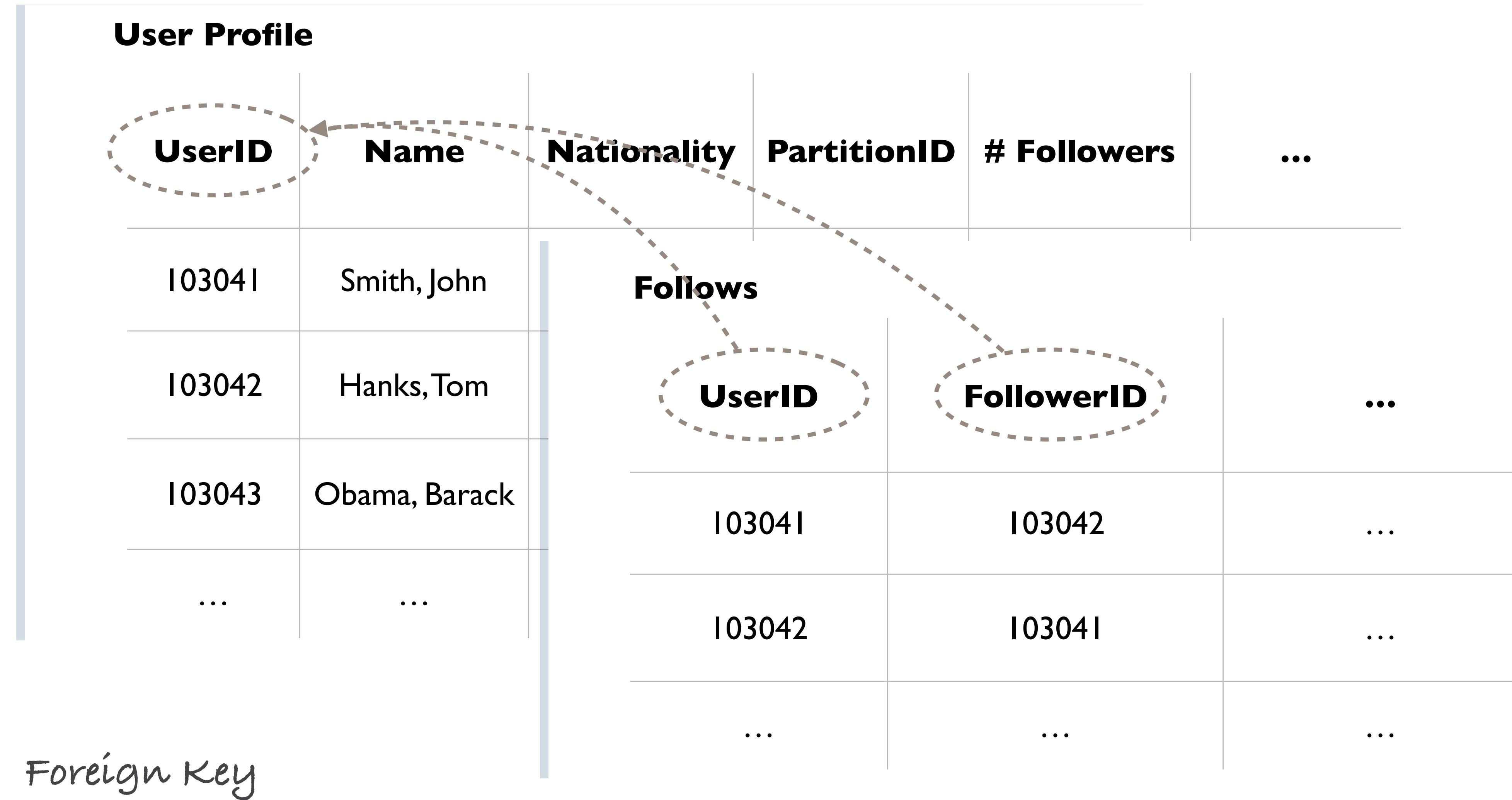
Integer

Relational Database Terms

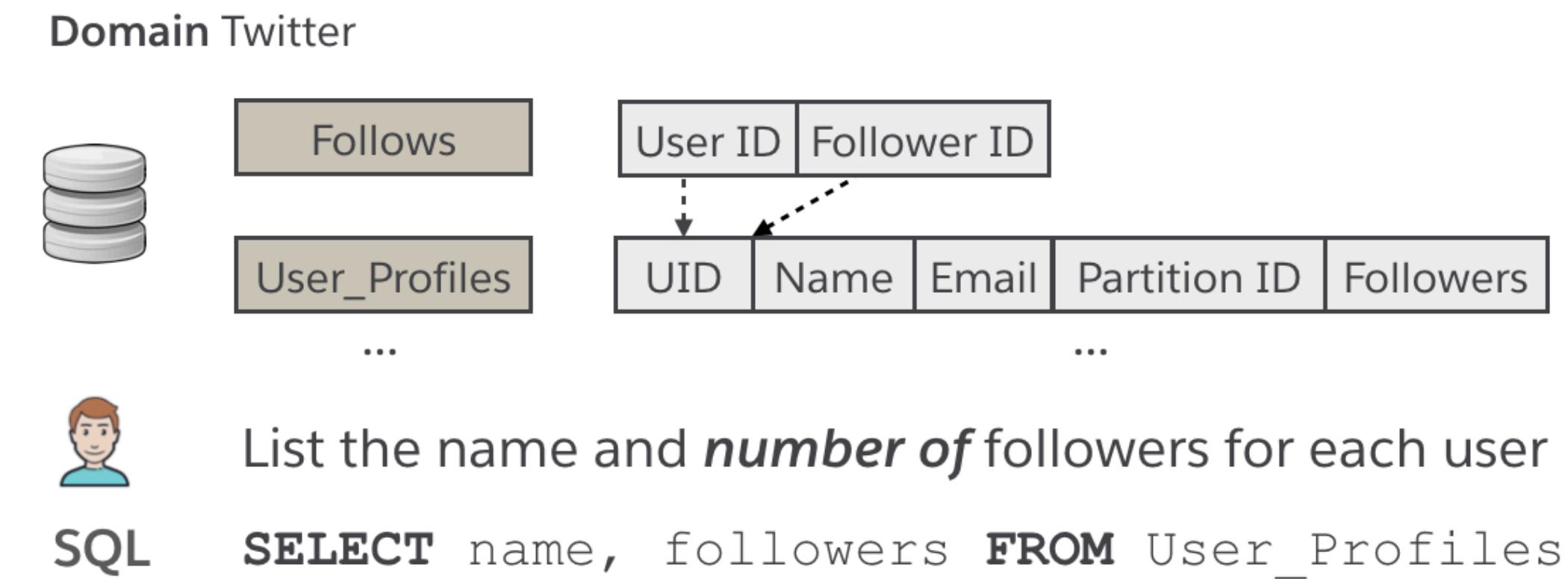
User Profile						
UserID	Name	Nationality	PartitionID	# Followers
103041	Smith, John	Canada	P10	3
103042	Hanks, Tom	United States	P11	16.6M
103043	Obama, Barack	United States	P11	127M
...

Primary Key

Relational Database Terms



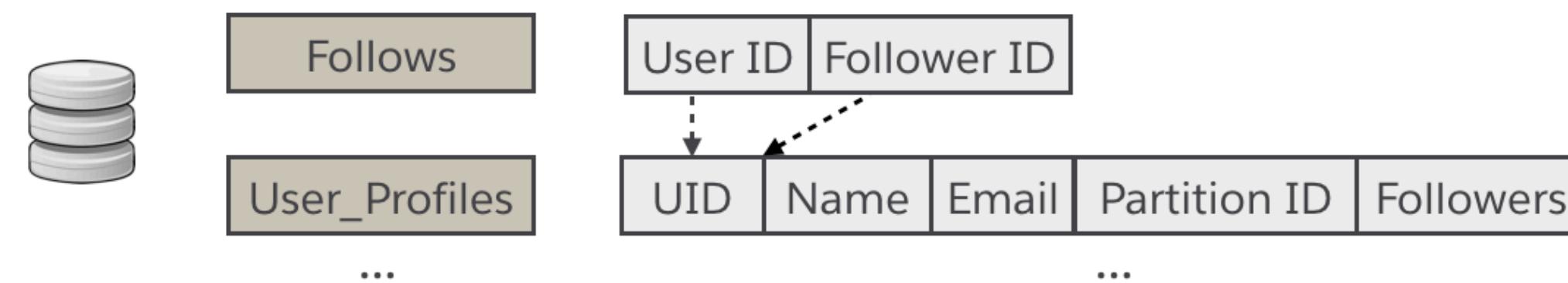
Text-to-SQL Semantic Parsing



Strong/Full
supervision

Cross Domain Text-to-SQL Semantic Parsing

Domain Twitter

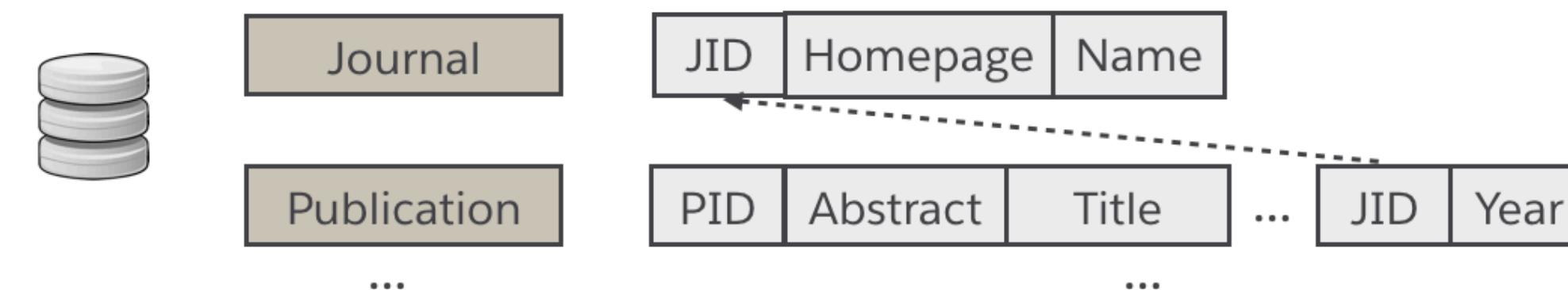


List the name and *number of* followers for each user

SQL

```
SELECT name, followers FROM User_Profiles
```

Domain Academic



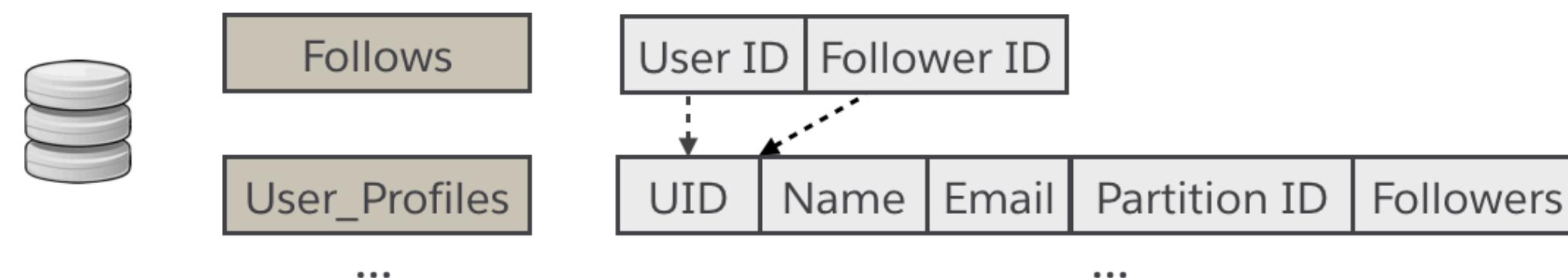
Return me the *number of* papers on PVLDB

SQL

```
SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = "PVLDB"
```

Challenges

Domain Twitter



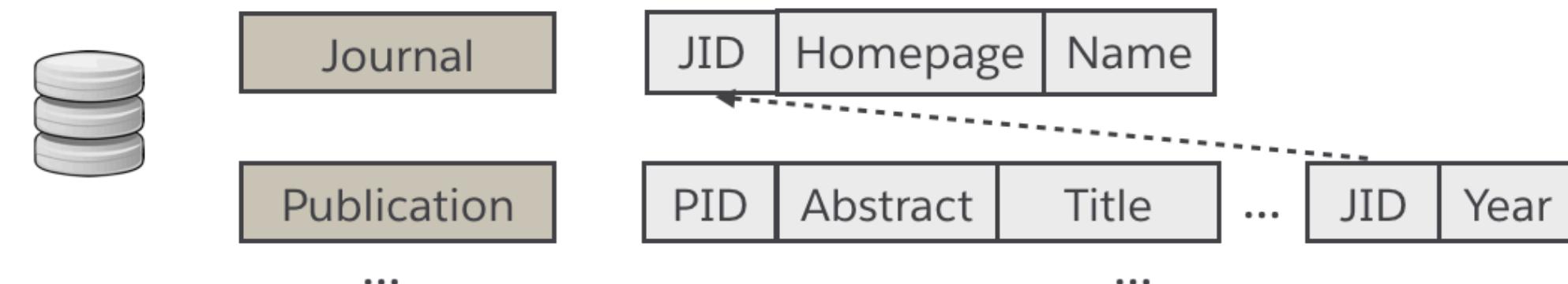
Challenge 1: Questions with similar intent may map to very different SQL logical forms when issued to different DBs.

>List the name and *number of* followers for each user

SQL `SELECT name, followers FROM User_Profiles`

Cross-
Database

Domain Academic



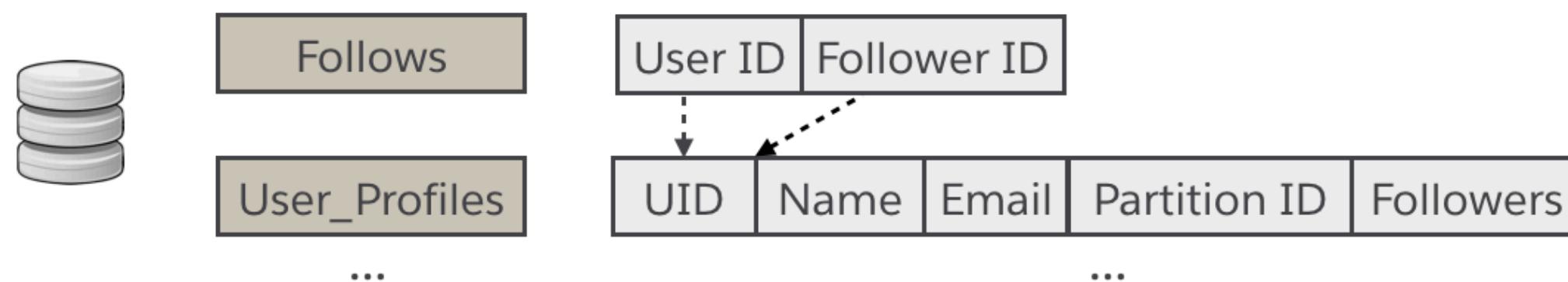
Return me the *number of* papers on PVLDB

SQL `SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = "PVLDB"`

Challenges

Challenge 2: Domain-specific entities are prevalent.

Domain Twitter

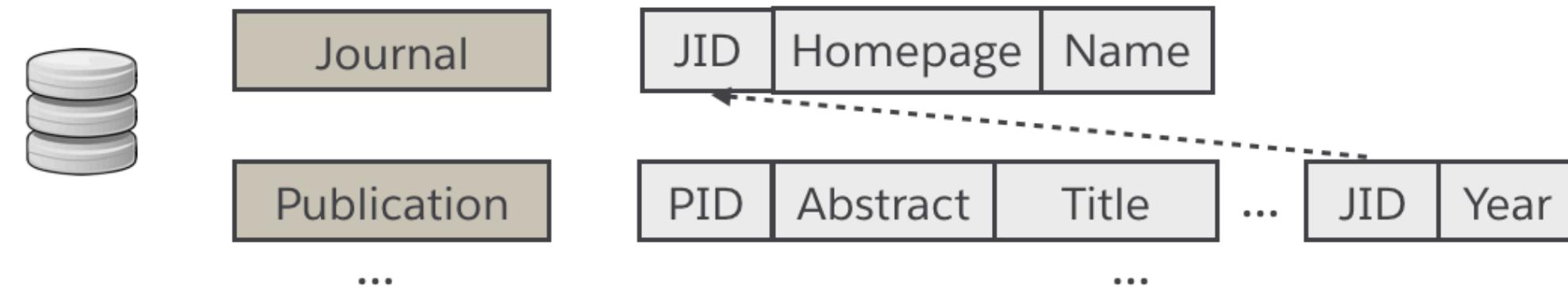


List the name and *number of* followers for each user

SQL

```
SELECT name, followers FROM User_Profiles
```

Domain Academic



Return me the *number of* papers on PVLDB

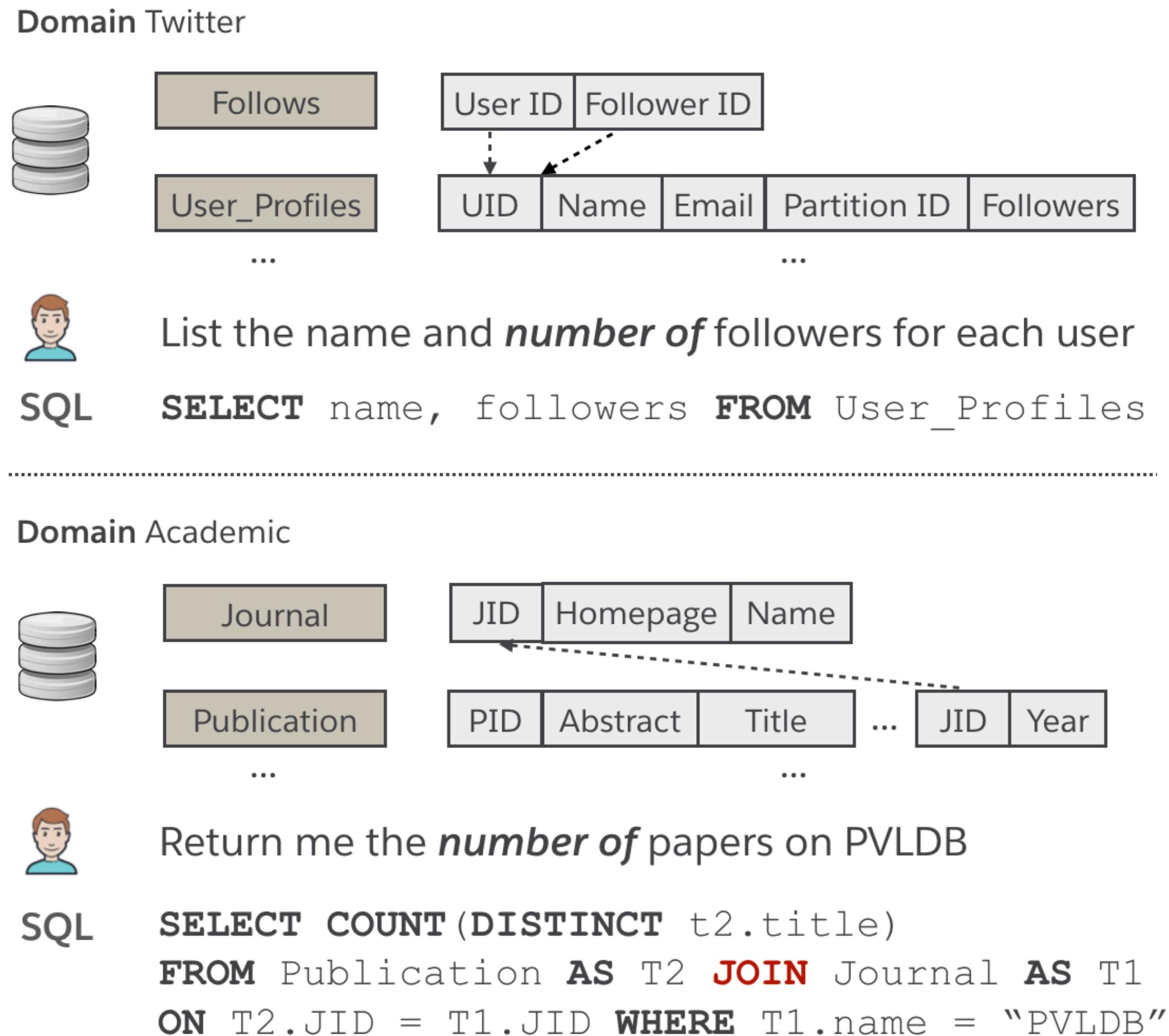
SQL

```
SELECT COUNT(DISTINCT t2.title)
FROM Publication AS T2 JOIN Journal AS T1
ON T2.JID = T1.JID WHERE T1.name = "PVLDB"
```

Domain Diversity

Challenges

-  **Observation:** Text-DB grounding is critical to this problem.
 - Model needs to recognize the **table field** and **table value** mentioned by the question.
 - The meaning of a **table field** are determined by its name (lexical information), the **table header context** and its **field values**.
 - Grounding **question to table values** also resolves the domain-specific entity understanding problem to a large extent.



Model

Research Question:

How can we learn a representation that effectively captures the language grounding of an input question, the DB schema and the DB content?

Question

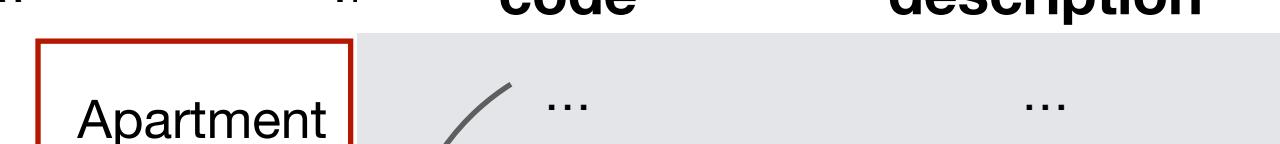
Database Schema

-  Show names of properties that are either houses or apartments

Properties					
Property id	Property type code	Property name	Date on market	Date sold	...
Apartment
Field
House
Shop
Other



Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...



Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

Joint Textual-Tabular Data Encoding

Serialize Table Header/DB Schema

Show names... Properties ... Property Type Code ... Reference Property Types ... Property Type Code ...

 Show names of properties that are either houses or apartments

Properties					
Property id	Property type code	Property name	Date on market	Date sold	...
Apartment
Field
House
Shop
Other

Reference Property Types	
Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

Joint Textual-Tabular Data Encoding

Serialize Table Header/DB Schema



 Show names of properties that are either houses or apartments

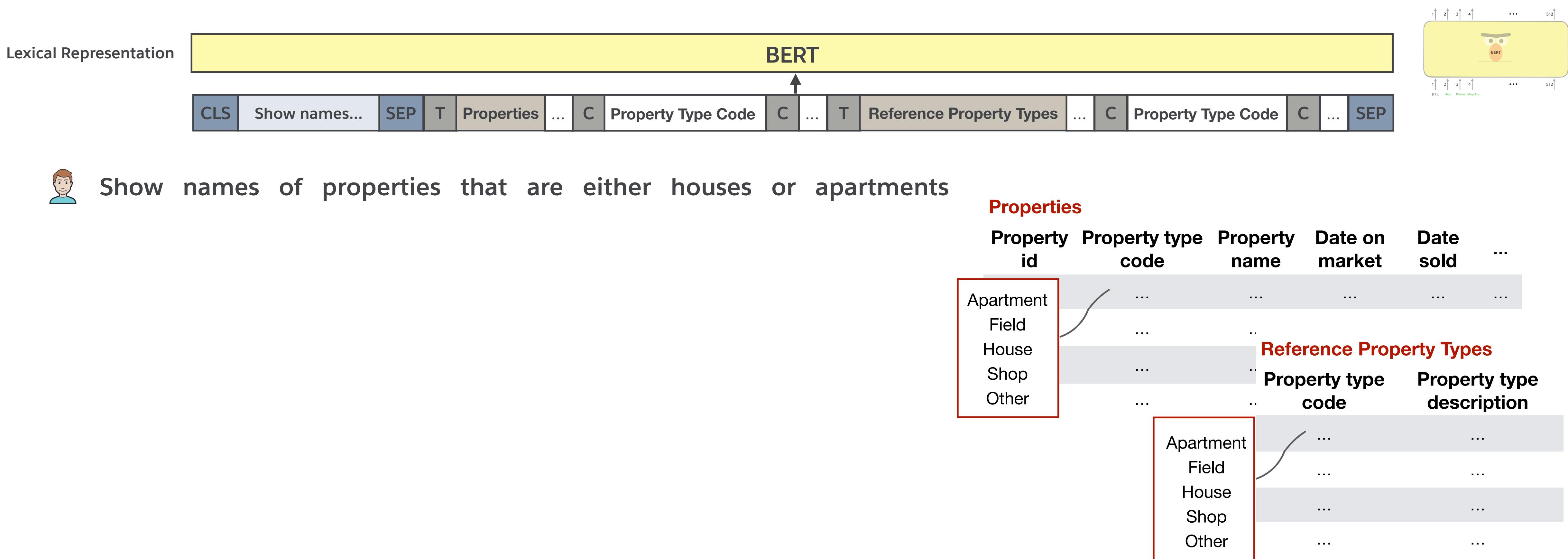
Properties	
Property id	Property type code
Apartment	...
Field	...
House	...
Shop	...
Other	...

 Reference Property Types

Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

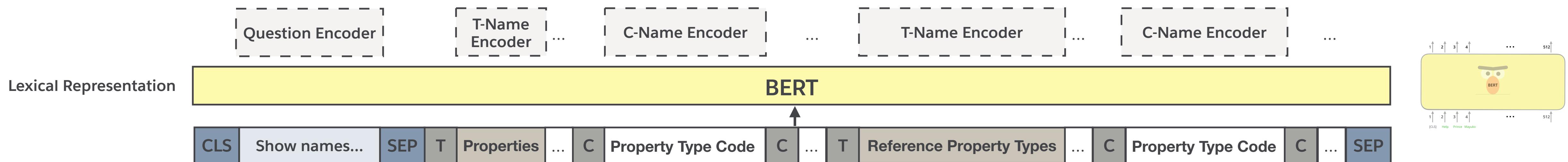
Joint Textual-Tabular Data Encoding

Serialize Table Header/DB Schema



Joint Textual-Tabular Data Encoding

Special-Purpose Layers



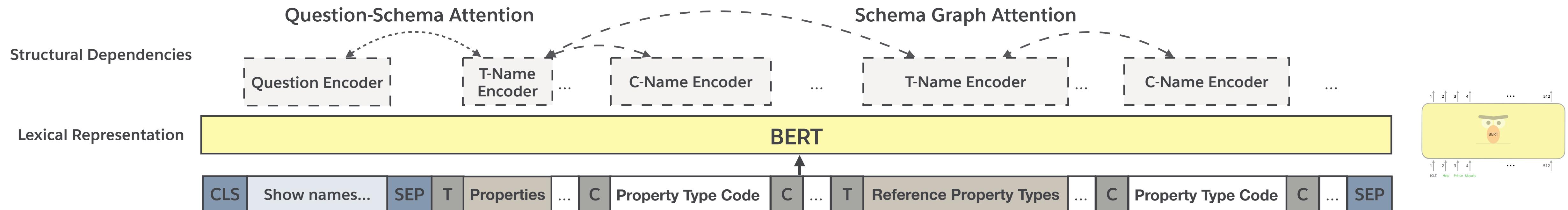
>Show names of properties that are either houses or apartments

Properties				
Property id	Property type code	Property name	Date on market	Date sold
Apartment
Field
House
Shop
Other

Reference Property Types	
Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

Joint Textual-Tabular Data Encoding

Intra/Inter Modality Attention



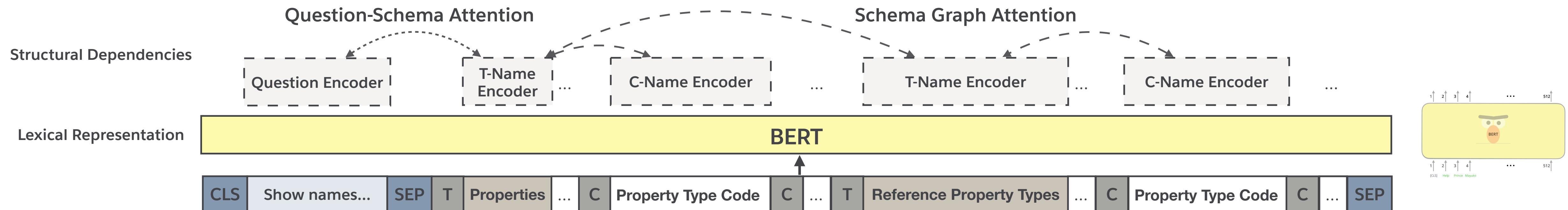
>Show names of properties that are either houses or apartments

Properties		Reference Property Types	
Property id	Property type code	Property name	Date on market
Apartment
Field
House
Shop
Other

Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

Joint Textual-Tabular Data Encoding

Intra/Inter Modality Attention



>Show names of properties that are either houses or apartments

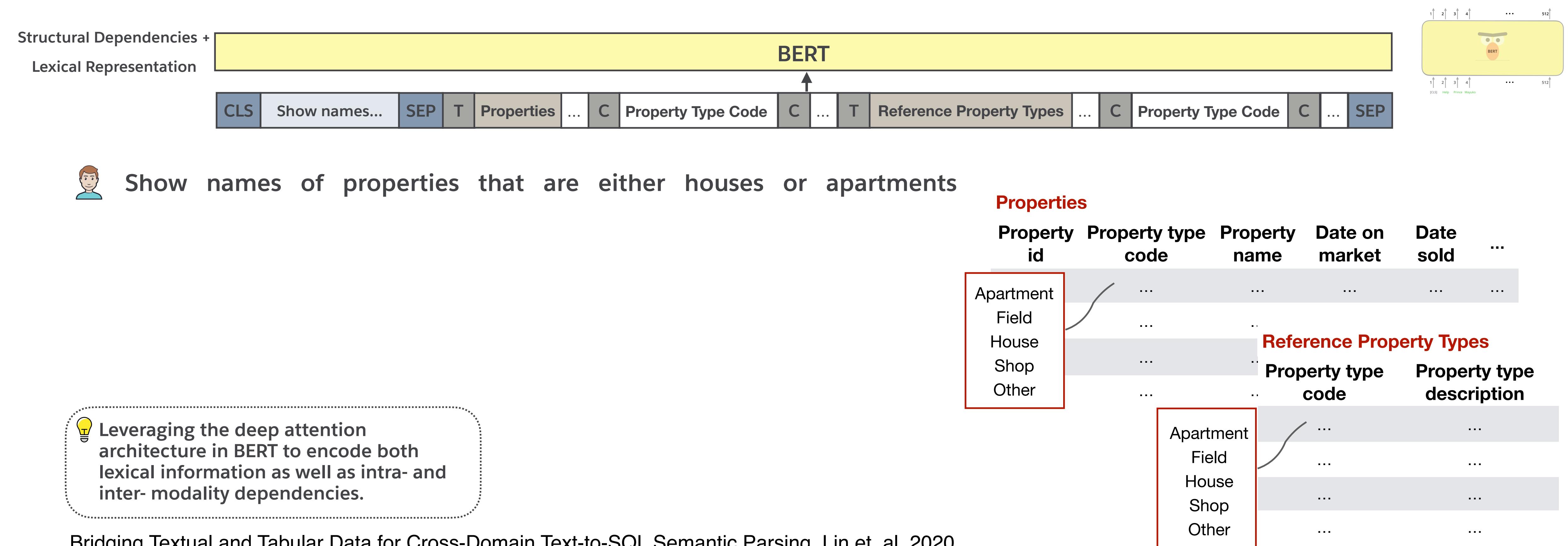
Properties					
Property id	Property type code	Property name	Date on market	Date sold	...
Apartment
Field
House
Shop
Other

Reference Property Types	
Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

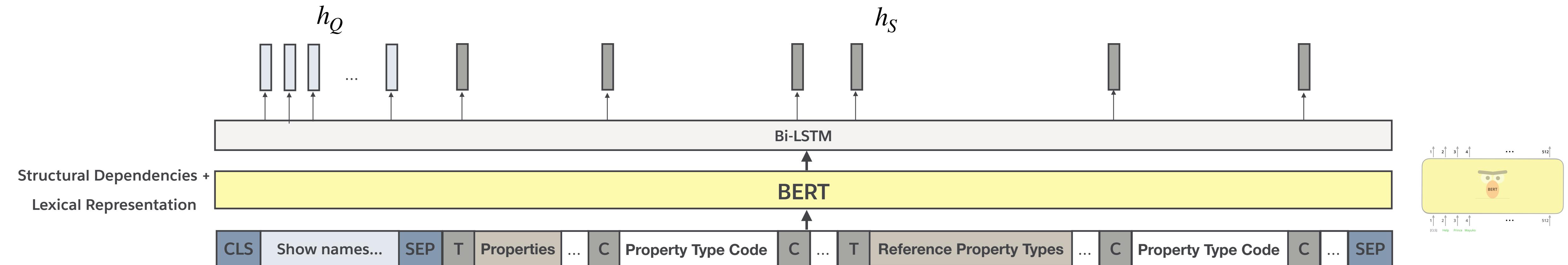
+6 Relational Self-Attention
Layers on top of BERT-large.
Model complexity quickly
increases.

Architecture redundancy

Joint Textual-Tabular Data Encoding



Joint Textual-Tabular Data Encoding



Show names of properties that are either houses or apartments

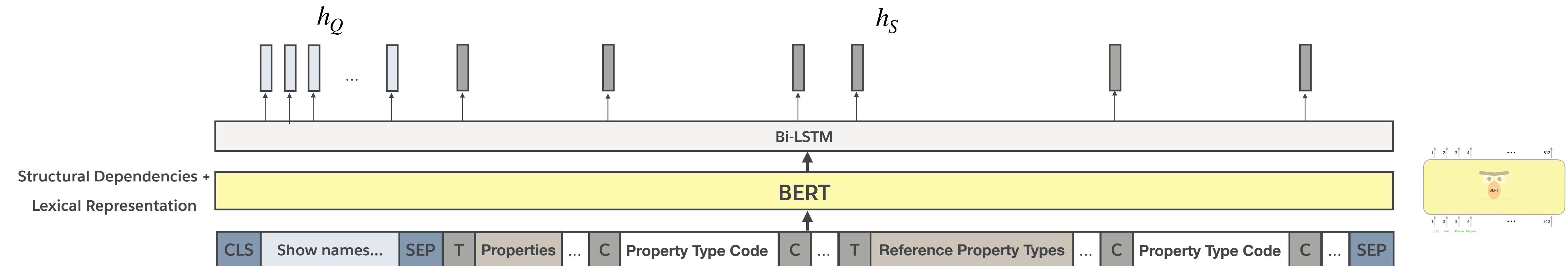
Properties

Property id	Property type code	Property name	Date on market	Date sold	...
Apartment
Field
House
Shop
Other

Reference Property Types	Property type code	Property type description
Apartment
Field
House
Shop
Other

Leveraging the deep attention architecture in BERT to encode both lexical information as well as intra- and inter-modality dependencies.

Joint Textual-Tabular Data Encoding



Show names of properties that are either houses or apartments

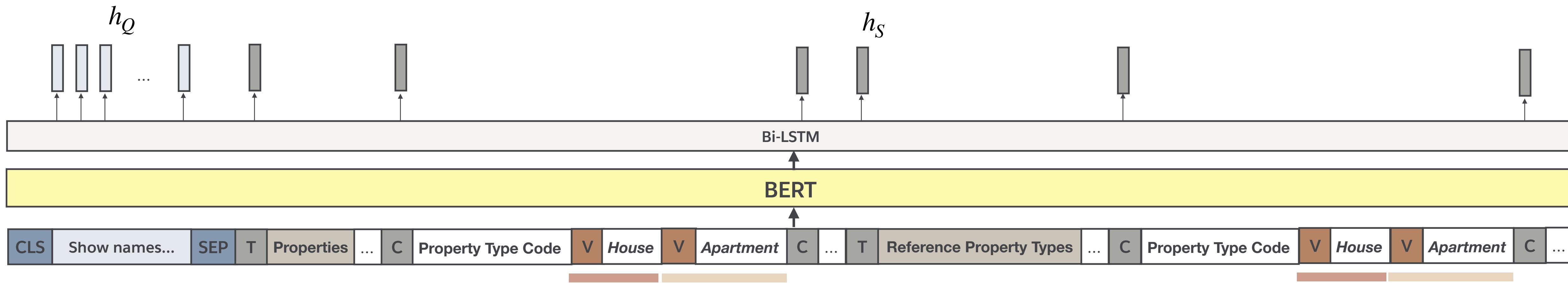
Properties

Property id	Property type code	Property name	Date on market	Date sold
Apartment
Field
House
Shop
Other

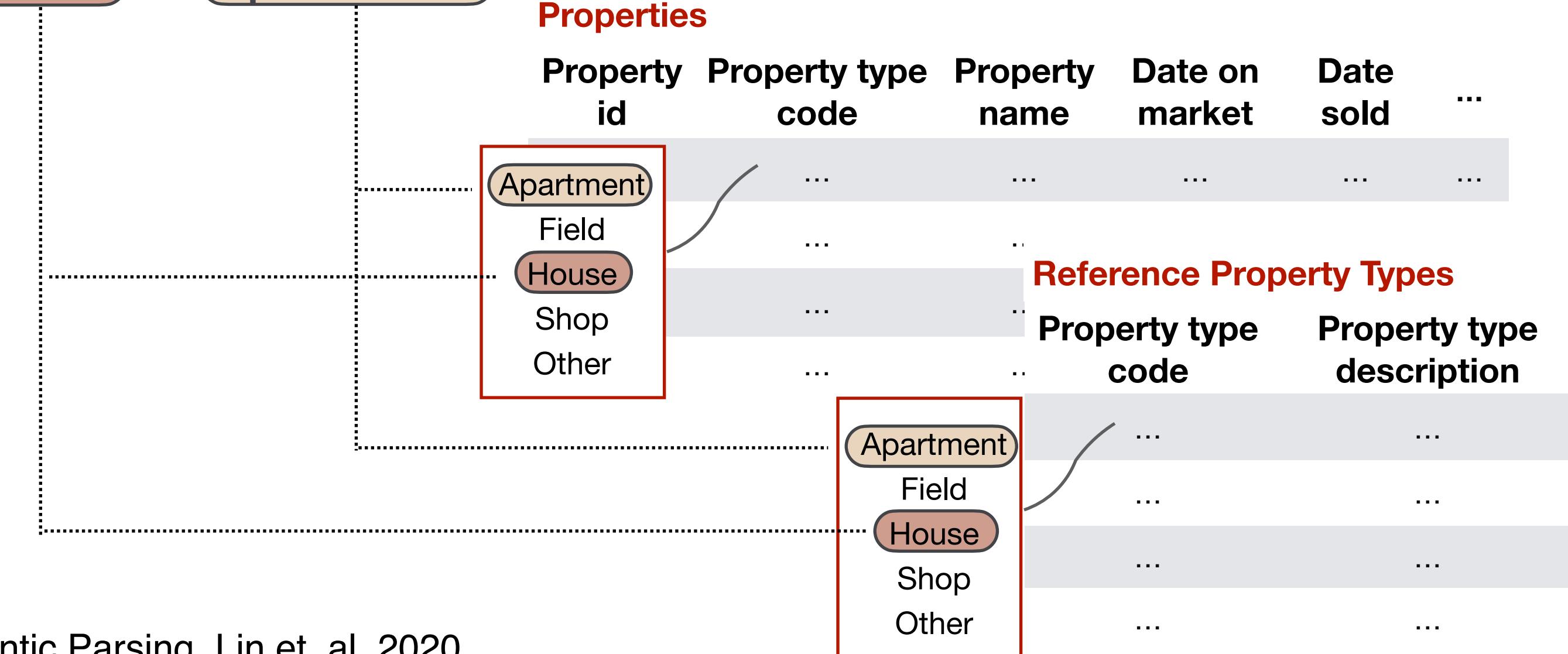
Reference Property Types

Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

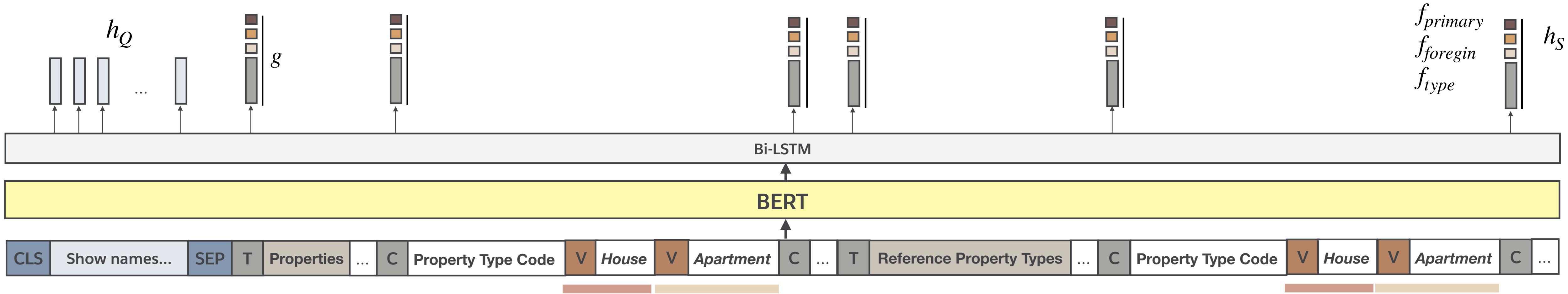
Bridging



 Show names of properties that are either houses or apartments



Meta-Data Encoding



Show names of properties that are either houses or apartments

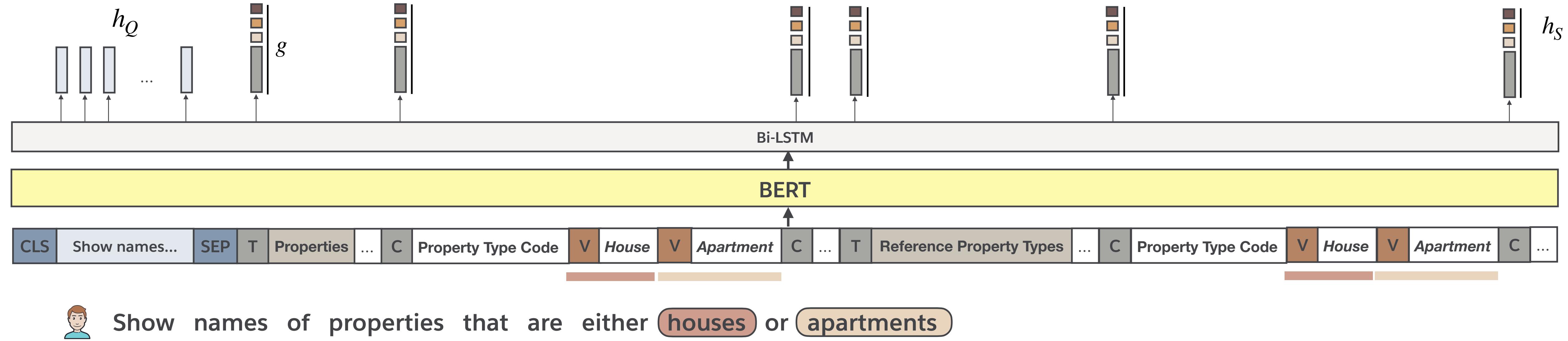
Properties

Property id	Property type code	Property name	Date on market	Date sold	...
Apartment
Field
House
Shop
Other

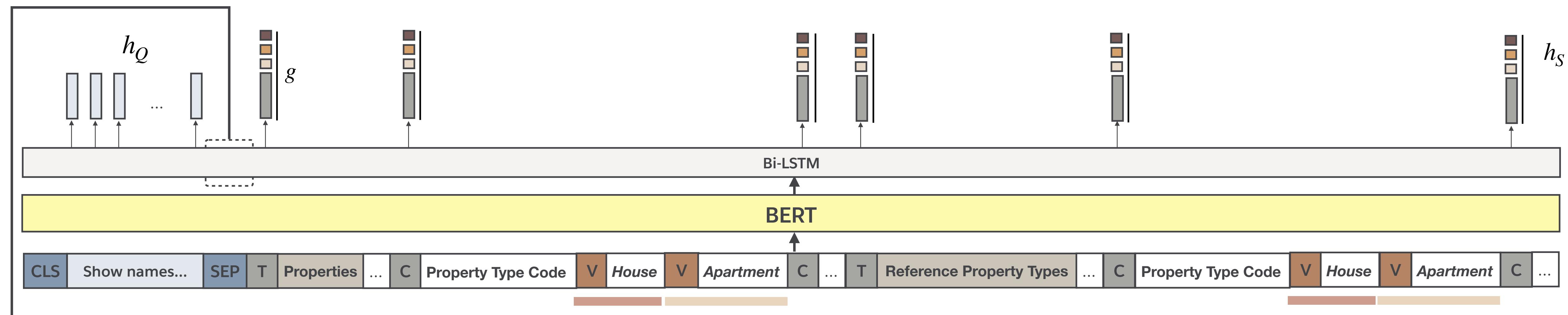
Reference Property Types

Property type code	Property type description
Apartment	...
Field	...
House	...
Shop	...
Other	...

Decoder

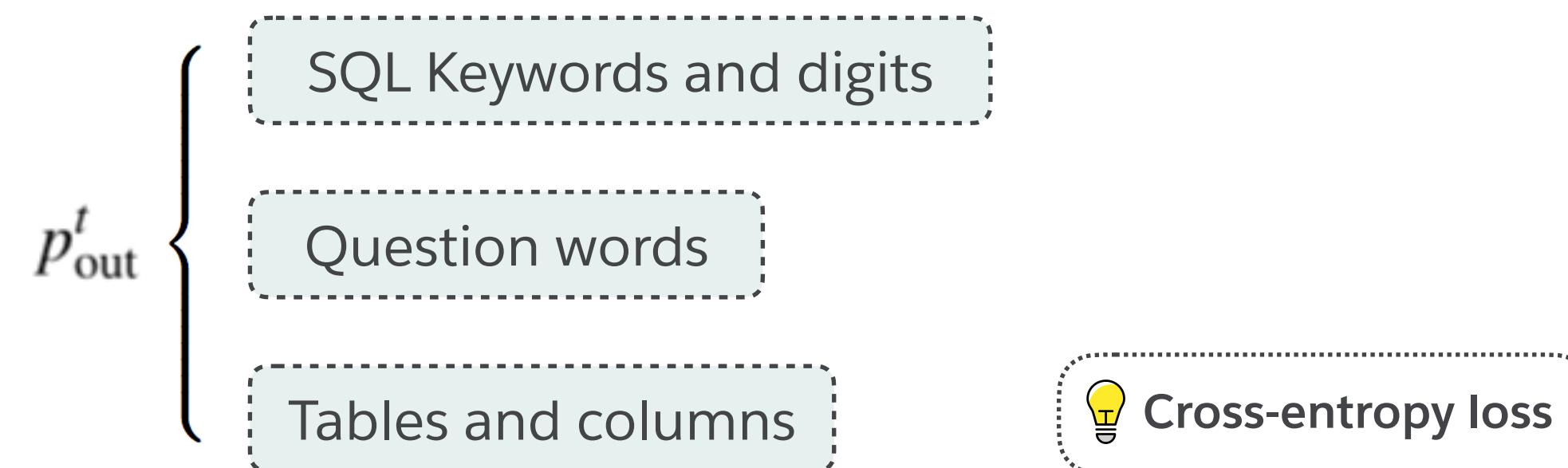


Decoder



>Show names of properties that are either houses or apartments

→LSTM-based pointer-generator (See et al. 2017)



Schema-Consistency Guided Decoding

Pruning the search space of a sequential pointer-generator decoder

- **Observation:** The FROM clauses set the scope of a SQL query and the table fields appeared in the rest of the clauses can only belong to the tables in FROM

```
SELECT T2.name FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID  
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (SELECT AVG(Rating) FROM Instructor)
```

Schema-Consistency Guided Decoding

Pruning the search space of a sequential pointer-generator decoder

- **Observation:** The FROM clauses set the scope of a SQL query and the table fields appeared in the rest of the clauses can only belong to the tables in FROM

```
SELECT T2.name FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID  
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (SELECT AVG(Rating) FROM Instructor)
```

 Rewrite a SQL query in execution order, with FROM clause at the beginning of each sub-query

```
FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID SELECT T2.name  
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (FROM Instructor SELECT AVG(Rating))
```

Schema-Consistency Guided Decoding



Pruning the search space of a sequential pointer-generator decoder

- **Observation:** The FROM clauses set the scope of a SQL query and the table fields appeared in the rest of the clauses can only belong to the tables in FROM

```
SELECT T2.name FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID  
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (SELECT AVG(Rating) FROM Instructor)
```

Rewrite a SQL query in execution order, with FROM clause at the beginning of each sub-query

```
FROM Instructor AS T1 JOIN Department AS T2 ON T1.Department_ID = T2.ID SELECT T2.name  
GROUP BY T1.Department_ID HAVING AVG(T1.Rating) > (FROM Instructor SELECT AVG(Rating))
```

Lemma: Let Y_{exec} be a SQL query with clauses arranged in execution order, then any table field in Y_{exec} will appear after its corresponding table token.

Schema-Consistency Guided Decoding



- Generate SQL queries in execution order and unmask DB fields dynamically



Schema-Consistency Guided Decoding



- Generate SQL queries in execution order and unmask DB fields dynamically



FROM

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



FROM Instructor

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



FROM Instructor JOIN

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



FROM Instructor JOIN Department

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



```
FROM Instructor JOIN Department ON Instructor.Department_ID = Department.ID SELECT  
Department.name GROUP BY Instructor.Department_ID HAVING_AVG(Instruction.Rating) >  
(FROM Instructor SELECT AVG(Instruction.Rating))
```

Schema-Consistency Guided Decoding

- Generate SQL queries in execution order and unmask DB fields dynamically



```
FROM Instructor JOIN Department ON Instructor.Department_ID = Department.ID SELECT  
Department.name GROUP BY Instructor.Department_ID HAVING_AVG(Instruction.Rating) >  
(FROM Instructor SELECT AVG(Instruction.Rating))
```

- ✓ Implemented via vector space computation
- ✓ Applied during inference only
- ✓ Cannot guarantee schema consistency, used in combination with static SQL correctness check
- ✓ Can be applied to other types of decoders

Dataset - Spider

Spider (Yu et al. 2018)

Expert-annotated, cross-domain, complex
text-to-SQL dataset

No overlap between Train/Dev/Test
databases, enabling the development of text-
to-SQL models which generalize to unseen DBs

	Train	Dev	Test
Hidden			
# DBs	146	20	40
# Examples	8,659	1,034	2,147

Database

Instructor

Primary key

ID	Name	Department_ID	Salary	...
----	------	---------------	--------	-----

Department

Primary key

ID	Name	Building	Budget	...
----	------	----------	--------	-----

...

Question What are the name and budget of the departments
with average instructor salary above the overall average?

SQL

```
SELECT T2.name, T2.budget
FROM Instructor AS T1 JOIN Department AS T2 ON
T1.Department_ID = T2.ID
GROUP BY T1.Department_ID
HAVING AVG(T1.salary) >
(SELECT AVG(Salary) FROM Instructor)
```

Dataset - WikiSQL

WikiSQL (Zhong et al. 2017)

Generated over **Wikipedia tables** using the **semantic-parsing-overnight** approach (Wang et al. 2015)

SQL Template: `SELECT $AGG $COLUMN
WHERE $COLUMN $OP $VALUE
(AND $COLUMN $OP $VALUE) *`

Train/Dev/Test tables overlap, but **49.6%** of dev tables are not in the train set and **45.1%** of test tables are not in the train set.

	Train	Dev	Test
# Tables	17,984	1,621	2,787
# Examples	56,355	8,421	15,878

WikiTable

Player	No.	Nationality	Position	Years in Toronto	School/Club

Question Who is the player that wears number 45?

SQL `SELECT Player WHERE No. = 42`

Experiment Setup

Pre-processing

- Compute fuzzy string match between the input question and the picklists of each DB field to obtain value mentions
- For each DB field, use the top- K matches in the DB schema representation ($K = 2$)

Decoding

- Beam search with length penalty
 - beam size = 16 for ablation study; beam size = 64 for leaderboard results

Evaluation

- Exact set match
 - Logical form match ignoring values and SQL component order invariance
- Execution accuracy
 - Check if the execution results of the predicted SQL query matches the executions results of the ground-truth SQL query

 Better evaluation for text-to-SQL is still an open research problem

Ablation Study

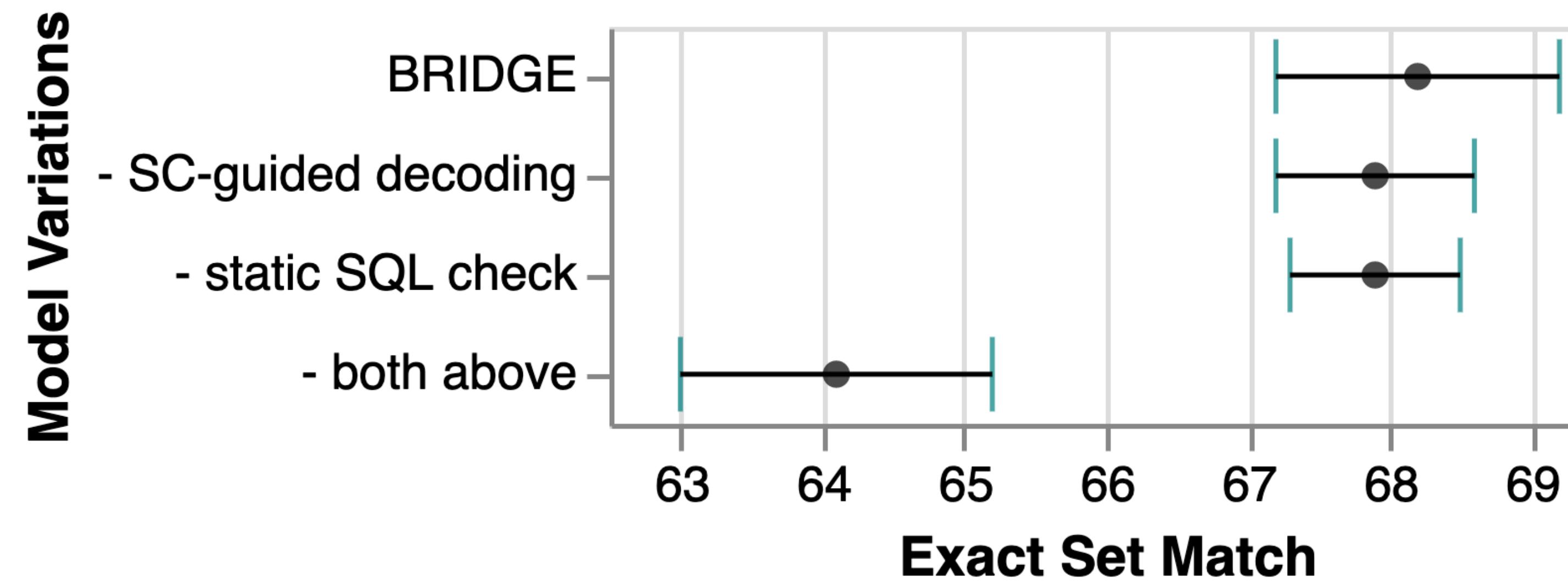


Figure 1. Ablation study of BRIDGE decoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

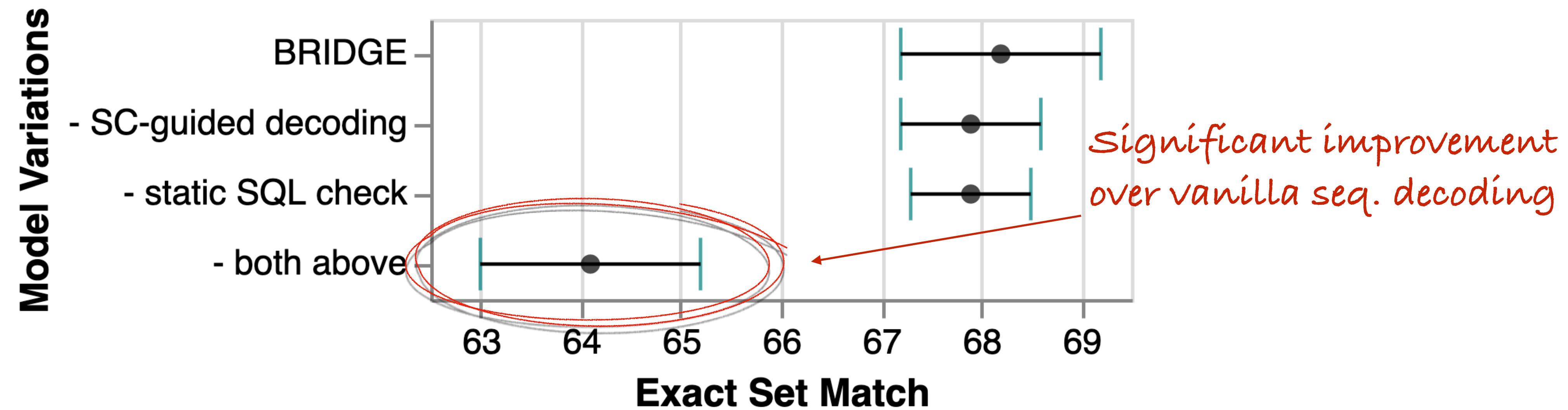


Figure 1. Ablation study of BRIDGE decoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

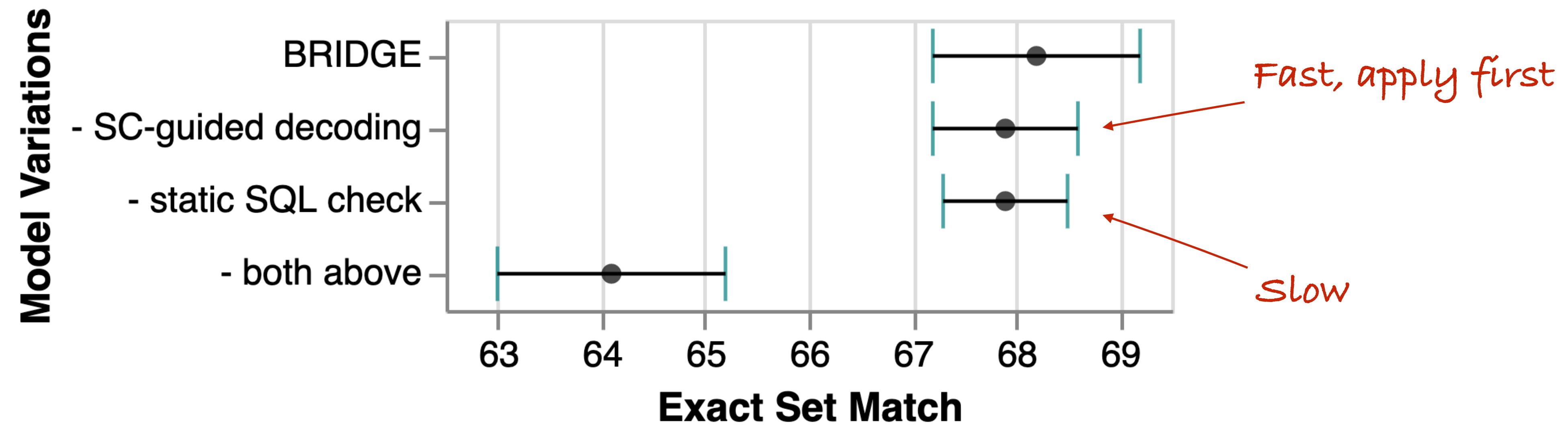


Figure 1. Ablation study of BRIDGE decoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

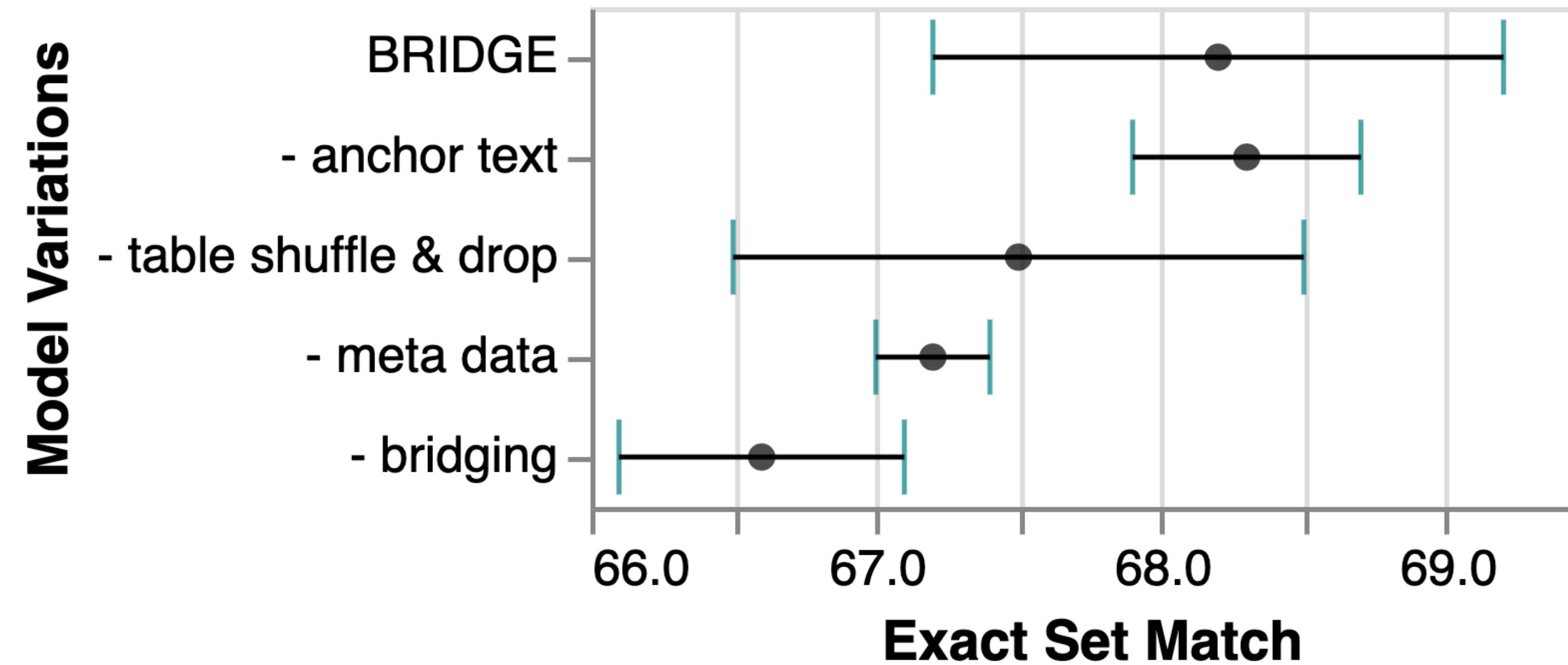


Figure 2. Ablation study of BRIDGE encoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

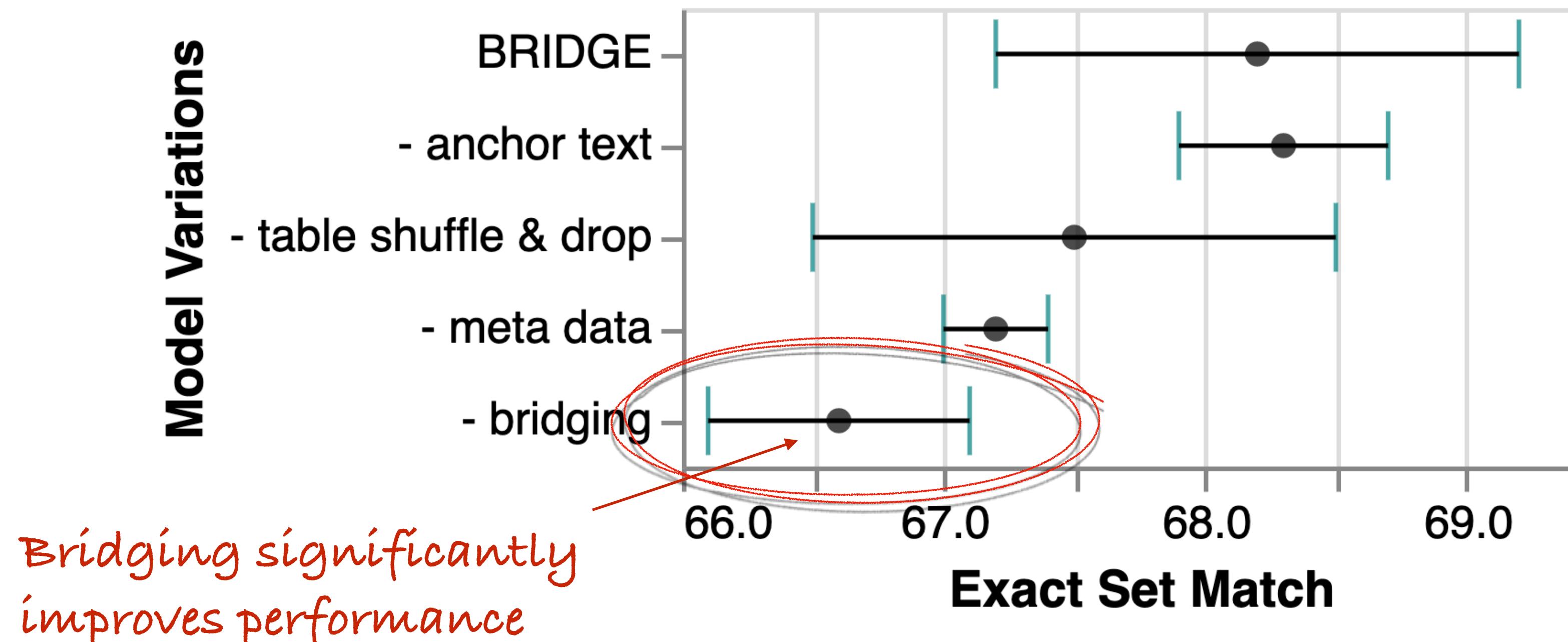


Figure 2. Ablation study of BRIDGE encoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

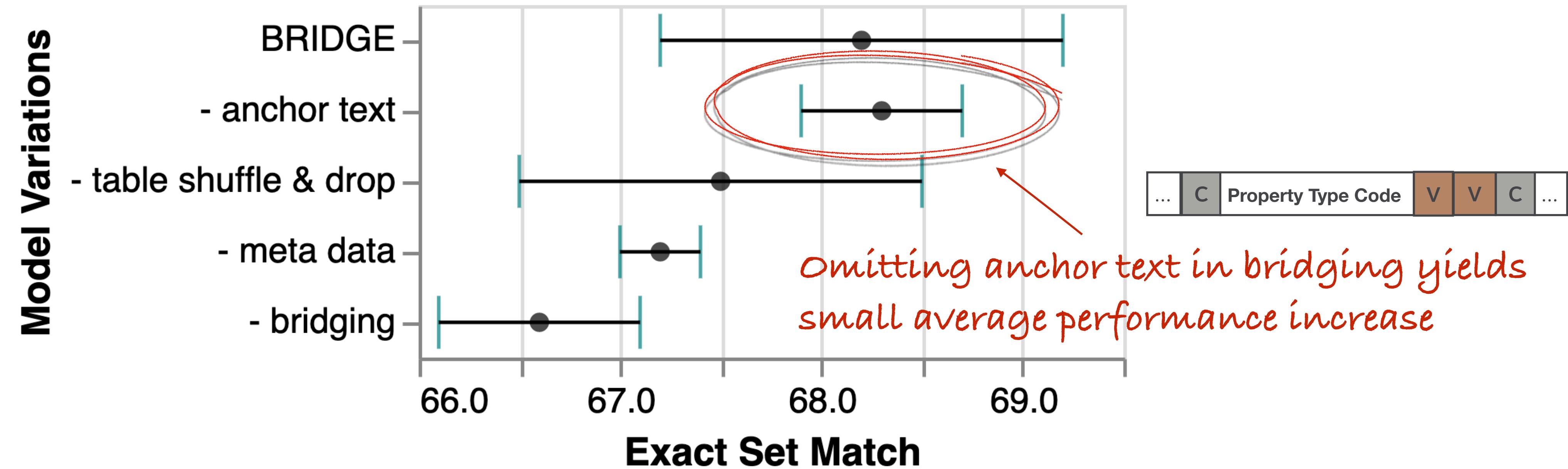


Figure 2. Ablation study of BRIDGE encoding strategies on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Bridging Ablation Performance by Difficulty Level

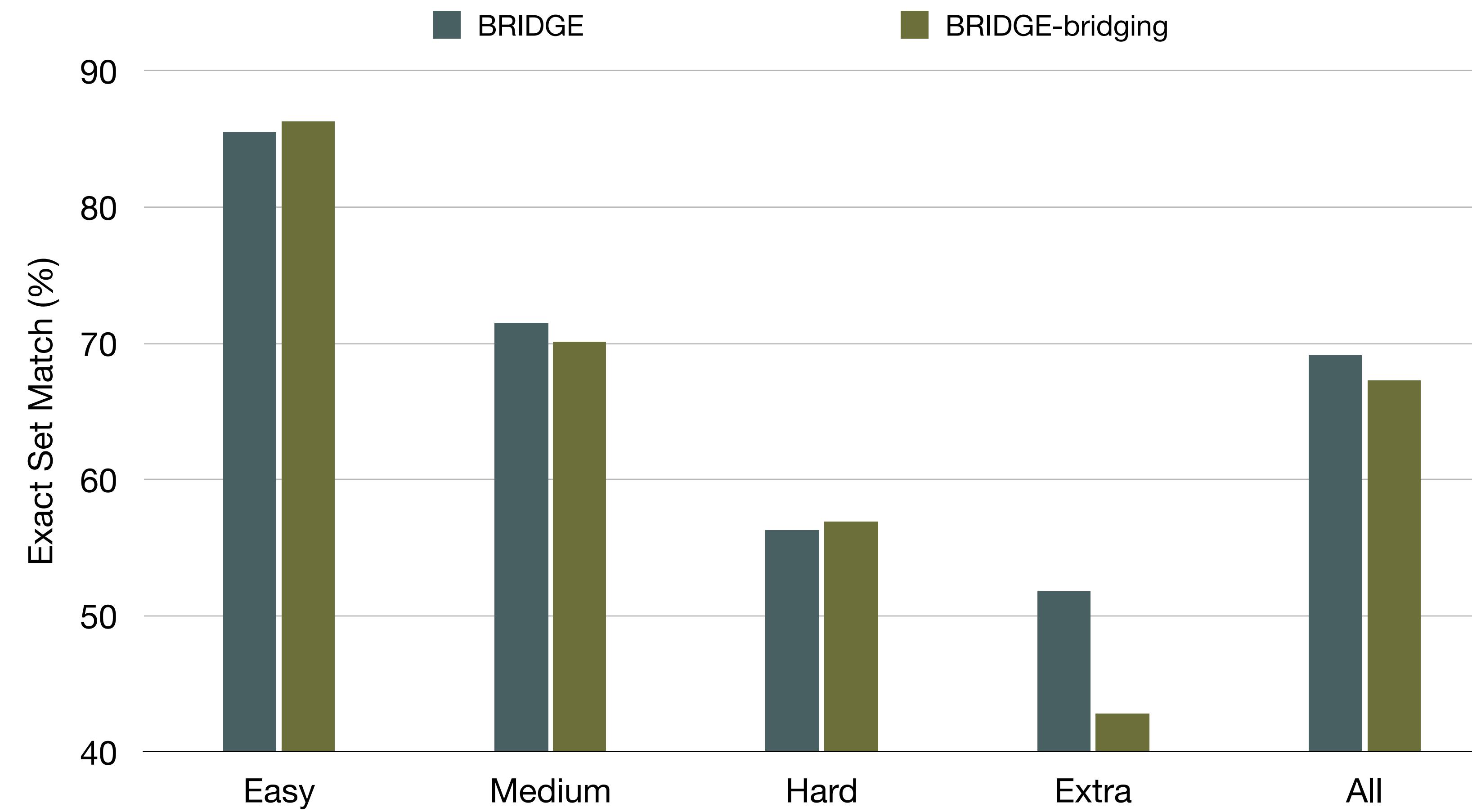


Figure 2.1. Performance of BRIDGE vs. BRIDGE - bridging on the Spider dev set.

Ablation Study

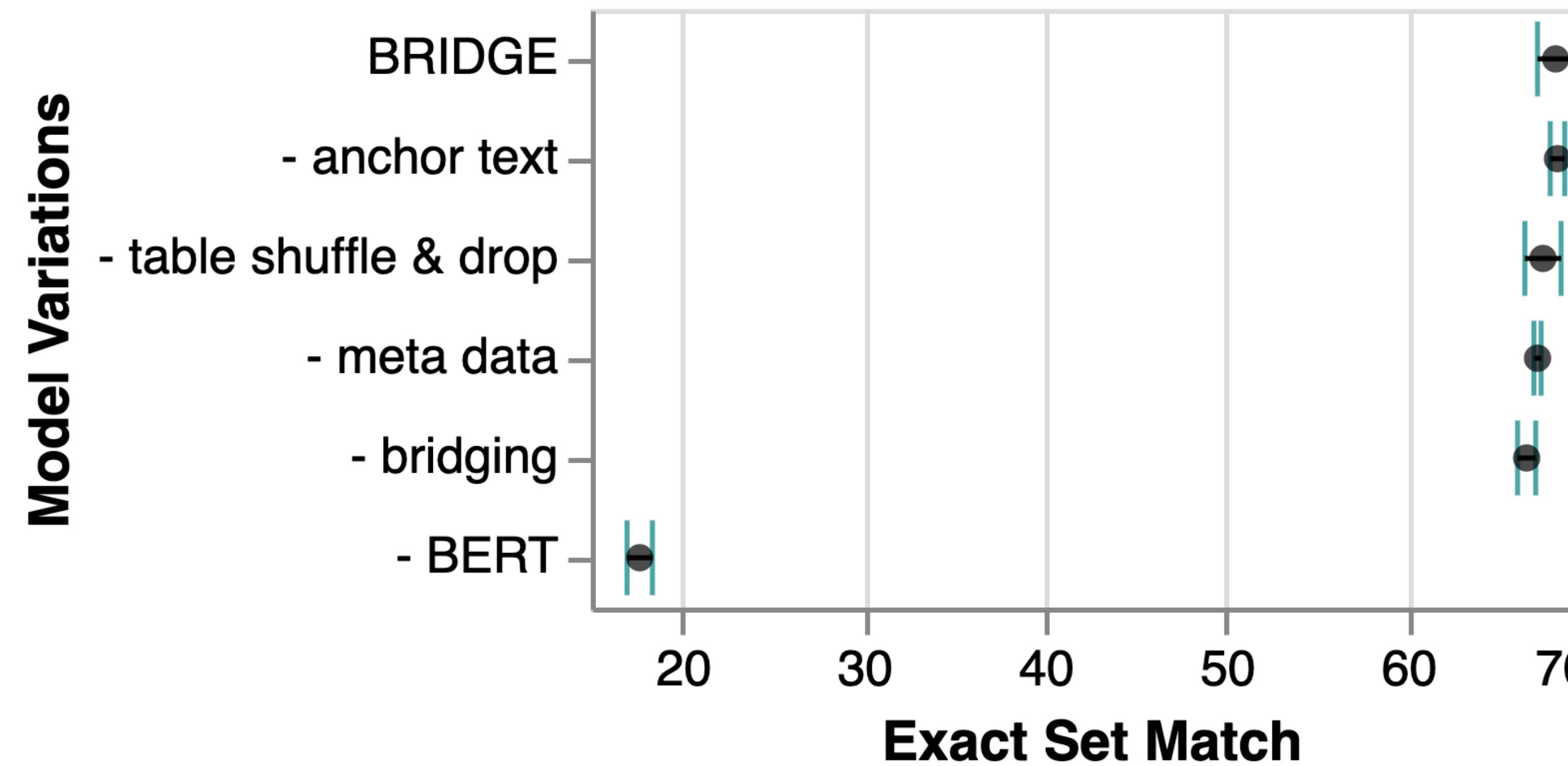
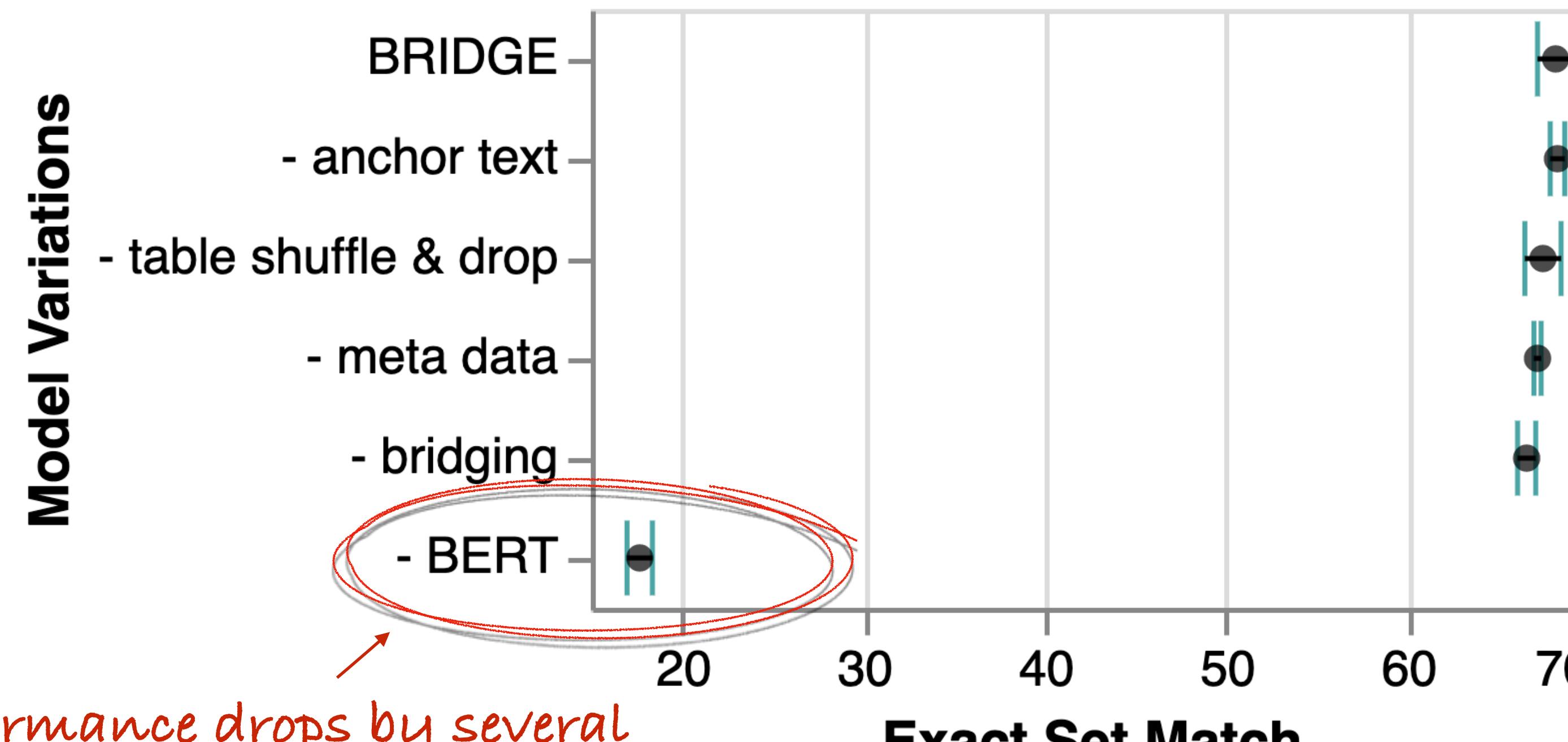


Figure 3. Ablation study of BRIDGE vs. BRIDGE - BERT on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study



Model performance drops by several folds if BERT encoding is not used

Figure 3. Ablation study of BRIDGE vs. BRIDGE - BERT on the Spider Dev set. We train 3 models using different random seeds for each model variation and average the performances.

Ablation Study

Model	w/o EG		w/ EG	
	EM	EX	EM	EX
BRIDGE _L	86.2	91.7	86.8	92.6
-anchor text	84.2	90.0	85.2	91.3
-bridging	82.6	88.5	84.5	90.8

Figure 4. Ablation study of BRIDGE WikiSQL Dev set. We train only 1 model for each model variation since model variation on WikiSQL is very small. EG refers to “execution guided decoding”.

Ensemble Model

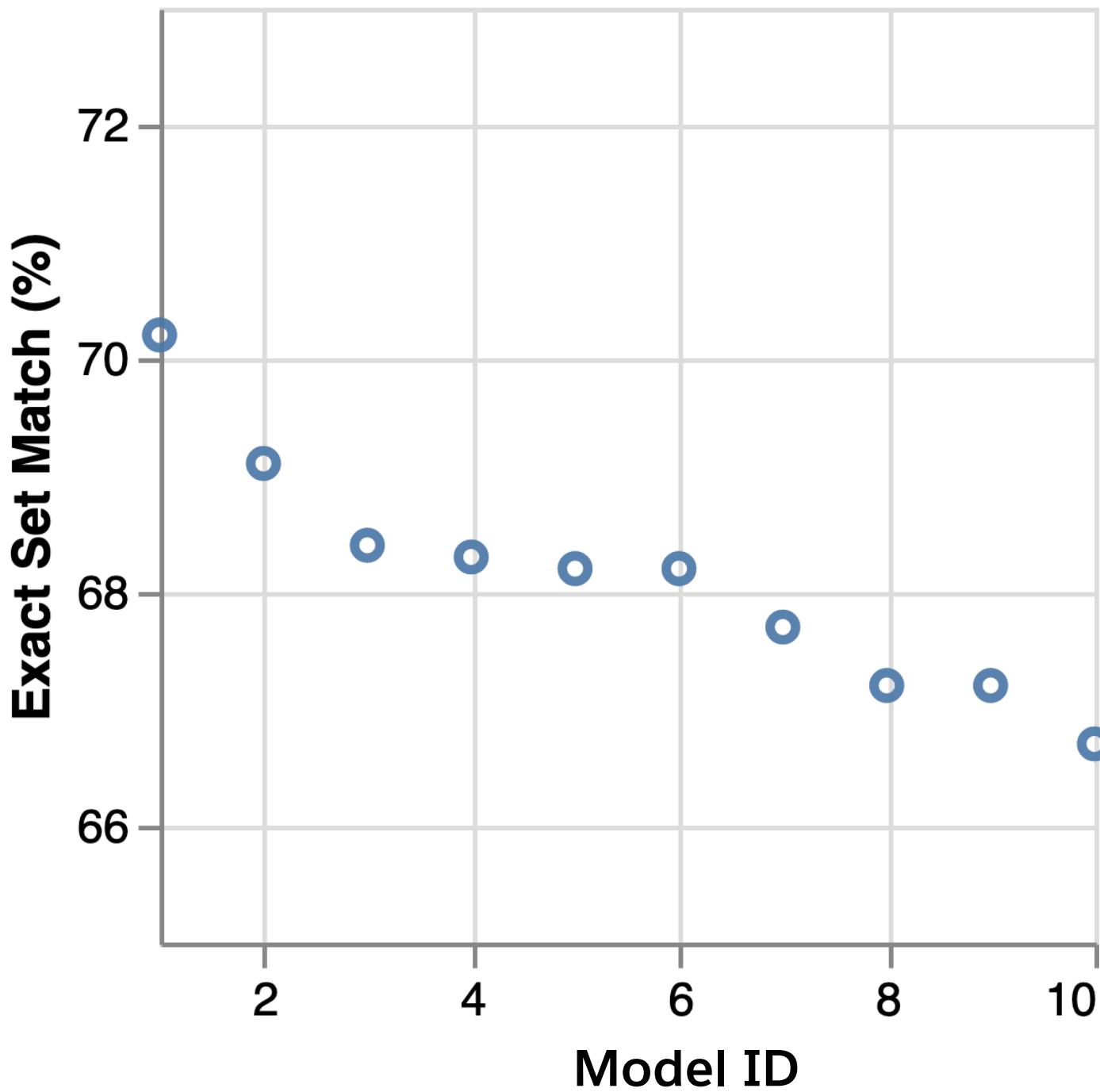


Figure 5. Performance comparison of BRIDGE model trained using 10 different random seeds on the Spider dev set.

Ensemble Model

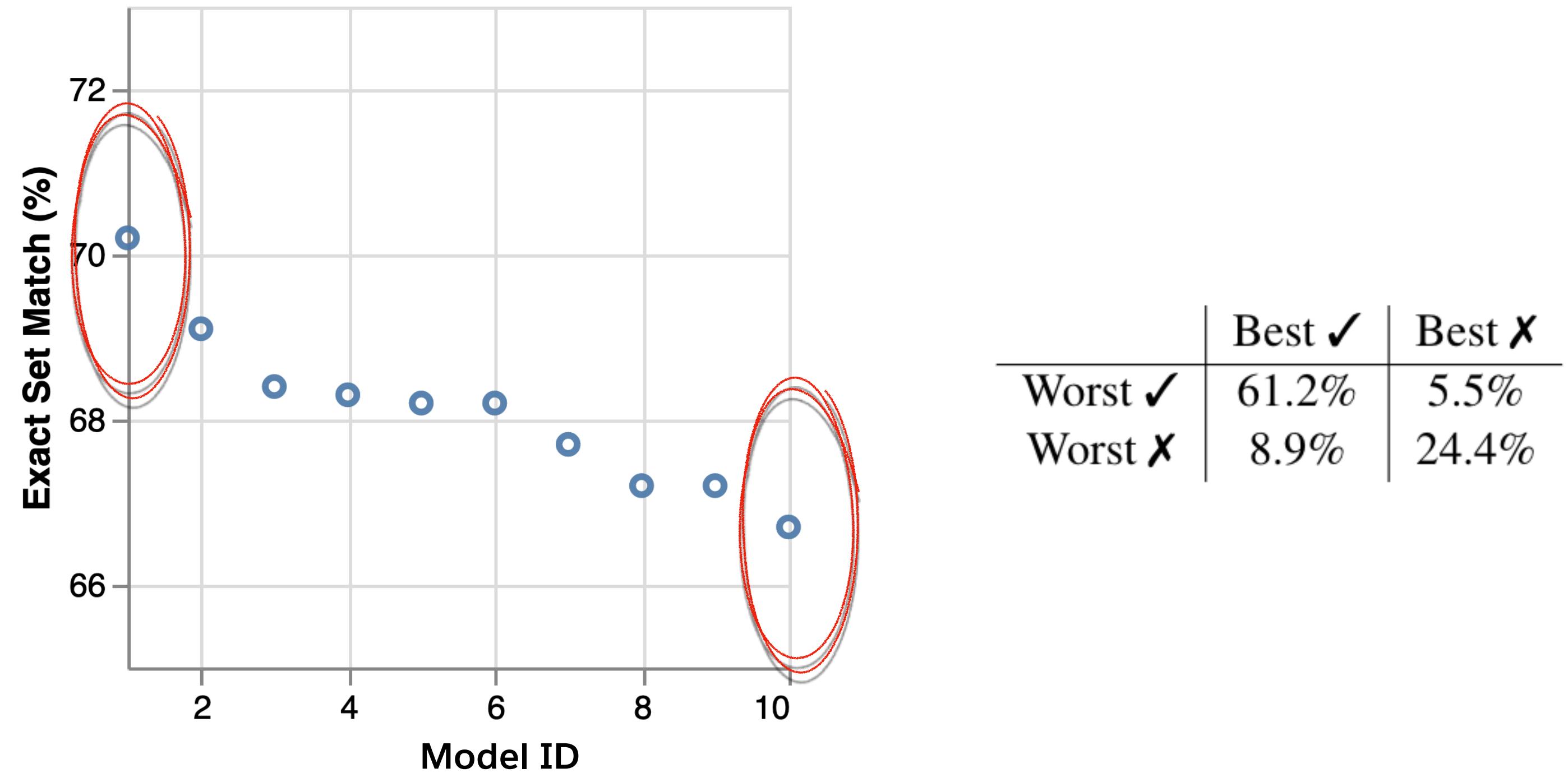


Figure 5. Performance comparison of BRIDGE model trained using 10 different random seeds on the Spider dev set.

Ensemble Model

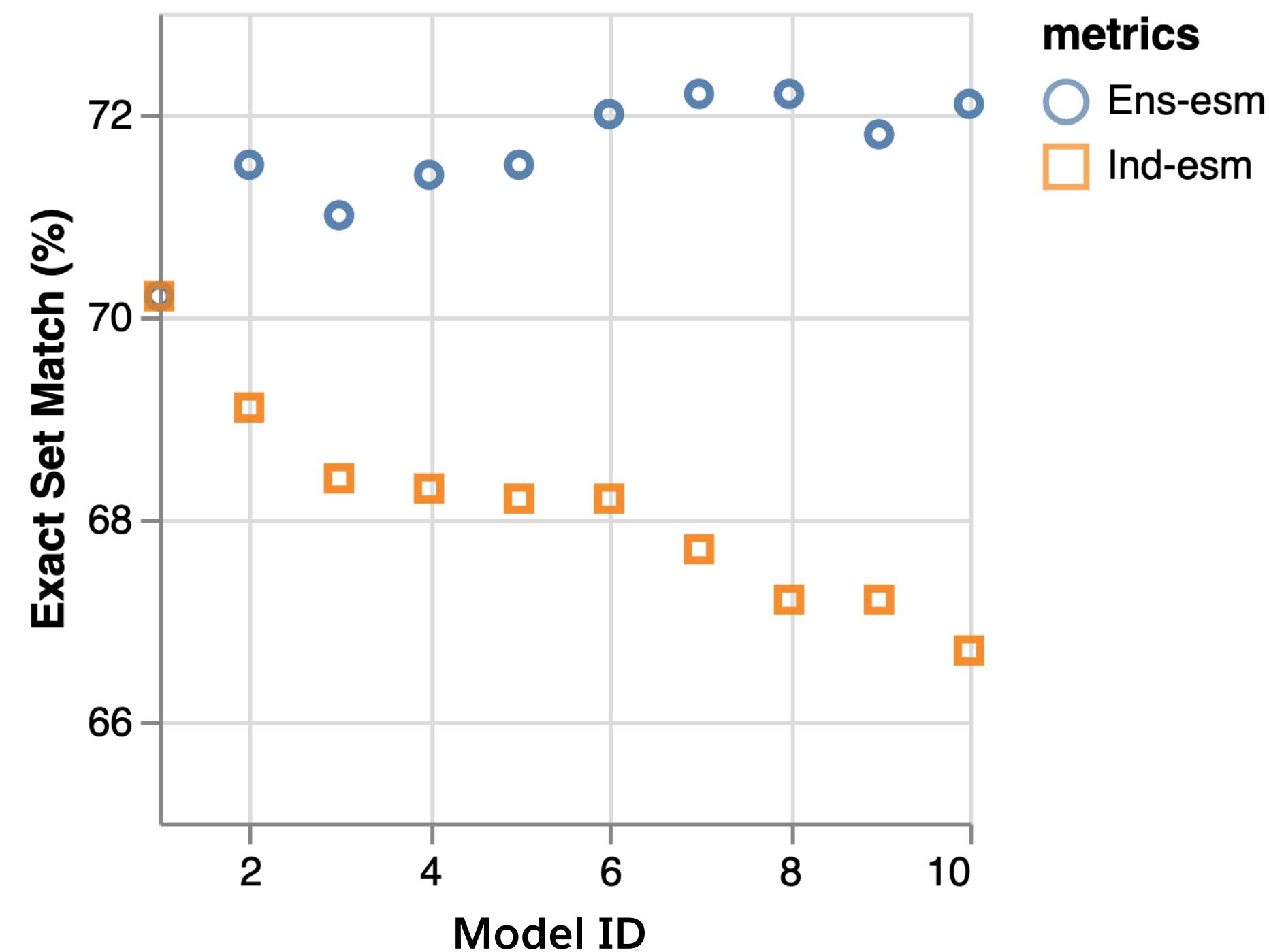
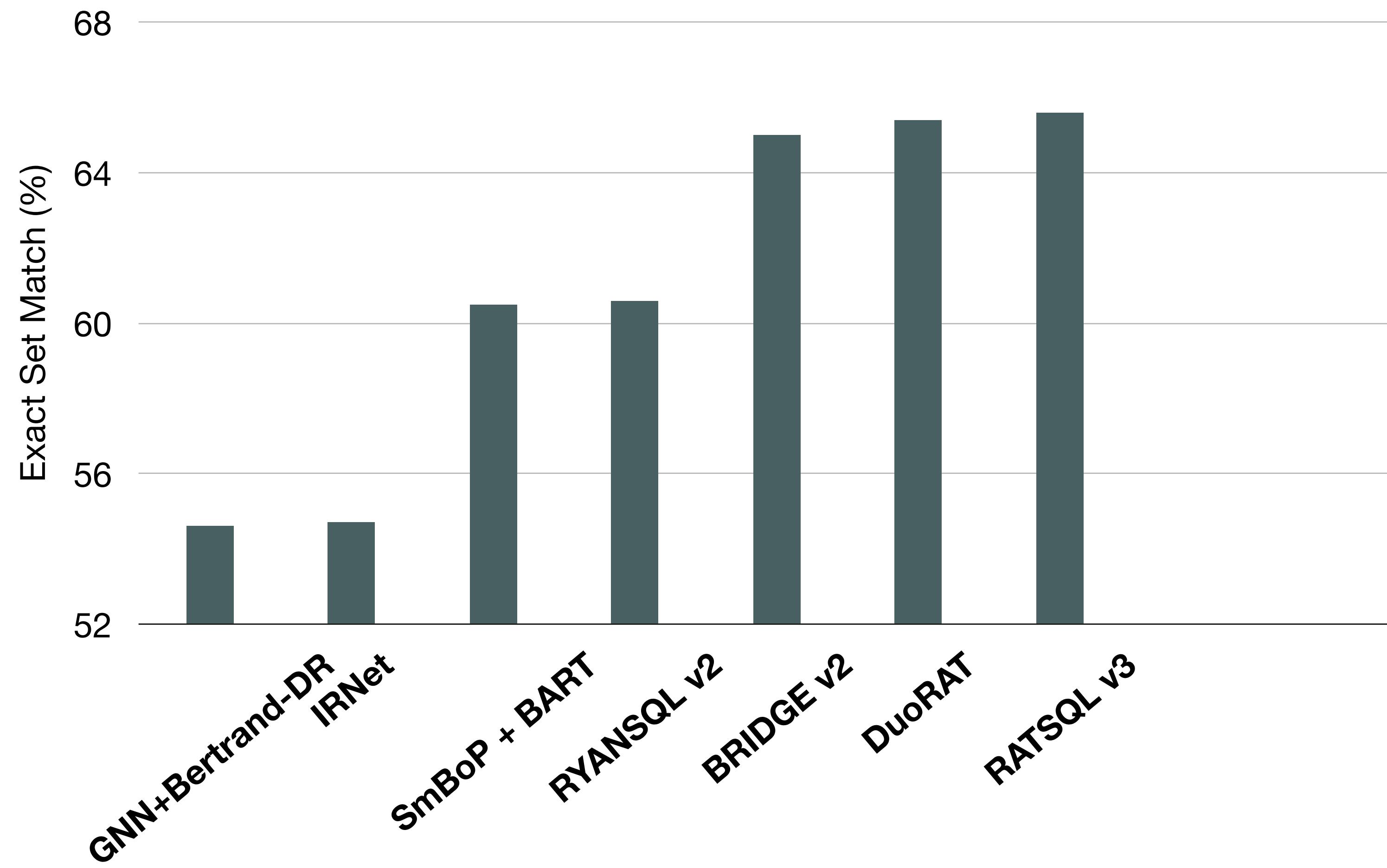


Figure 6. Performance of (step-wise average) model ensemble on the Spider dev set.

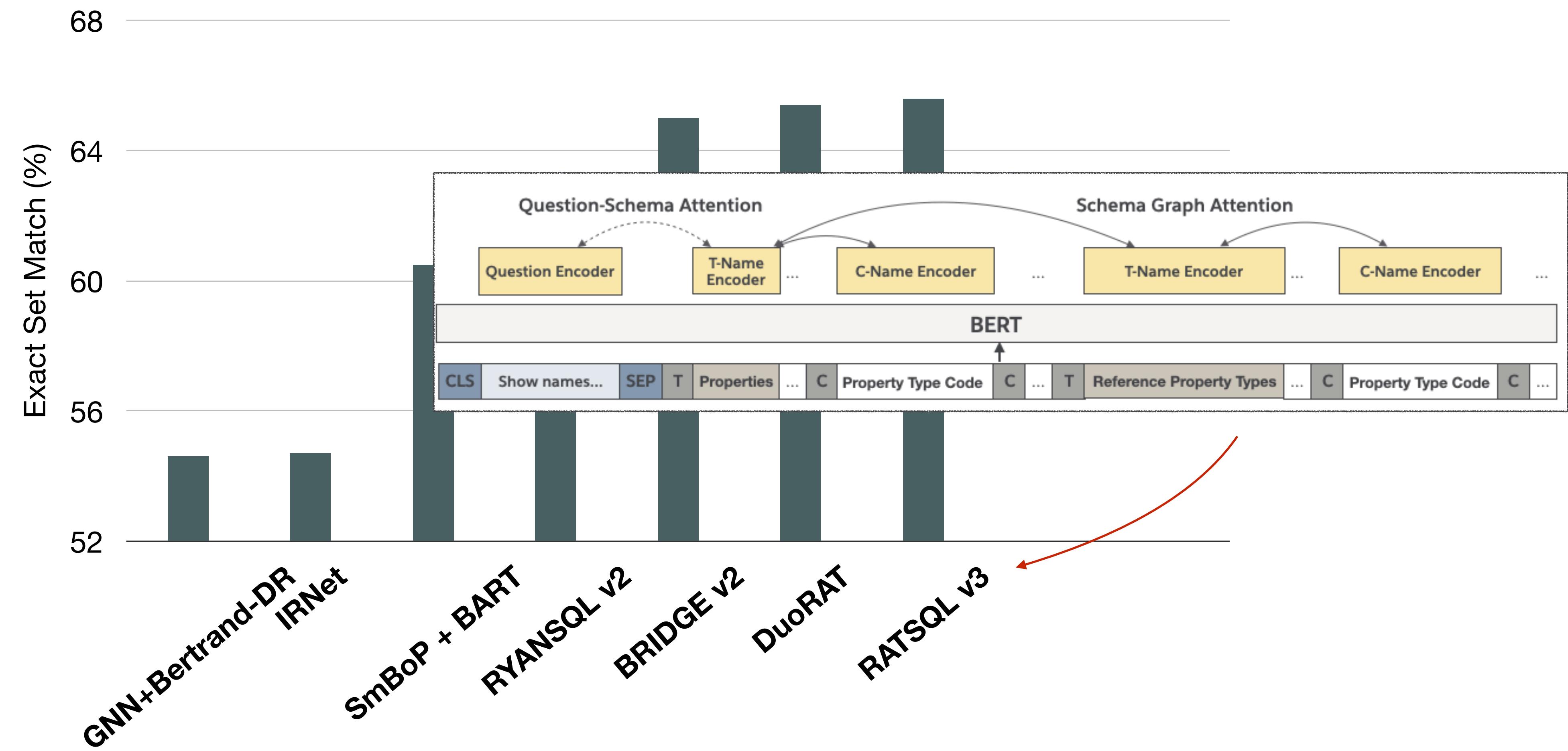
Performance on Spider Leaderboard

Figure 7. Comparison to other top-performing text-to-SQL models on the Spider leaderboard (Jan 31, 2020).



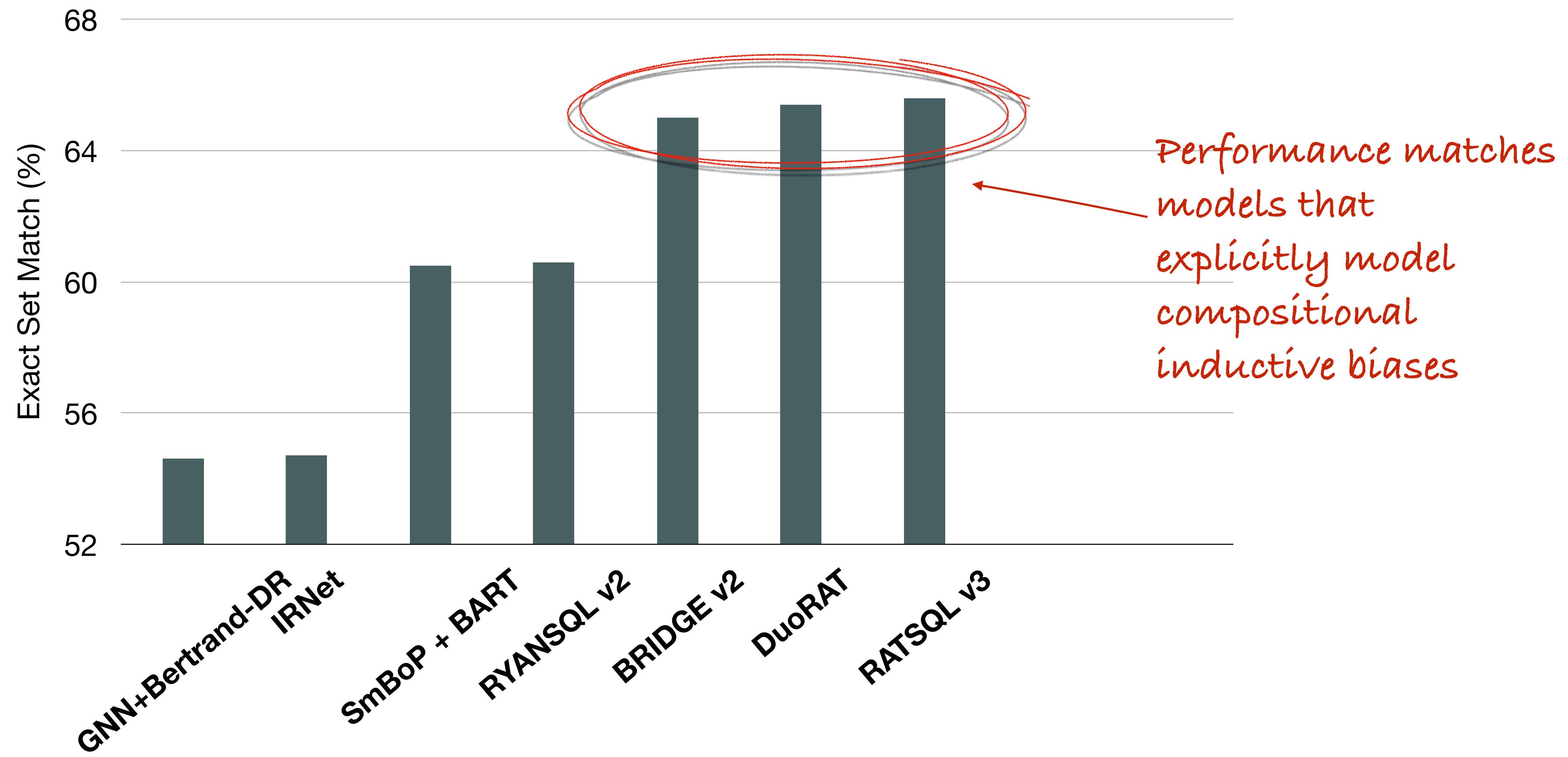
Performance on Spider Leaderboard

Figure 7. Comparison to other top-performing text-to-SQL models on the Spider leaderboard (Jan 31, 2020).



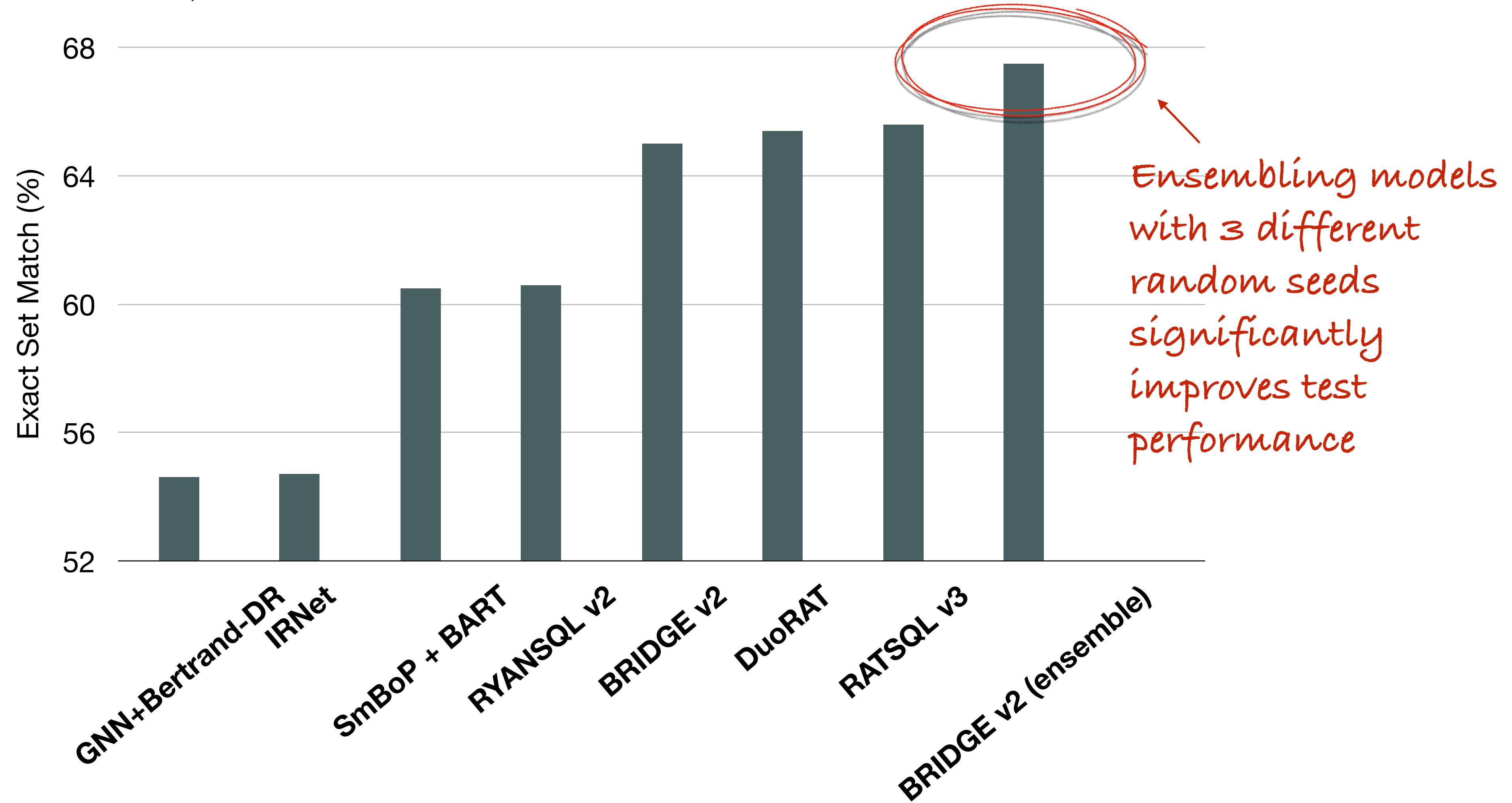
Performance on Spider Leaderboard

Figure 7. Comparison to other top-performing text-to-SQL models on the Spider leaderboard (Jan 31, 2020).



Performance on Spider Leaderboard

Figure 7. Comparison to other top-performing text-to-SQL models on the Spider leaderboard (Jan 31, 2020).



Performance Comparison by Difficulty Level

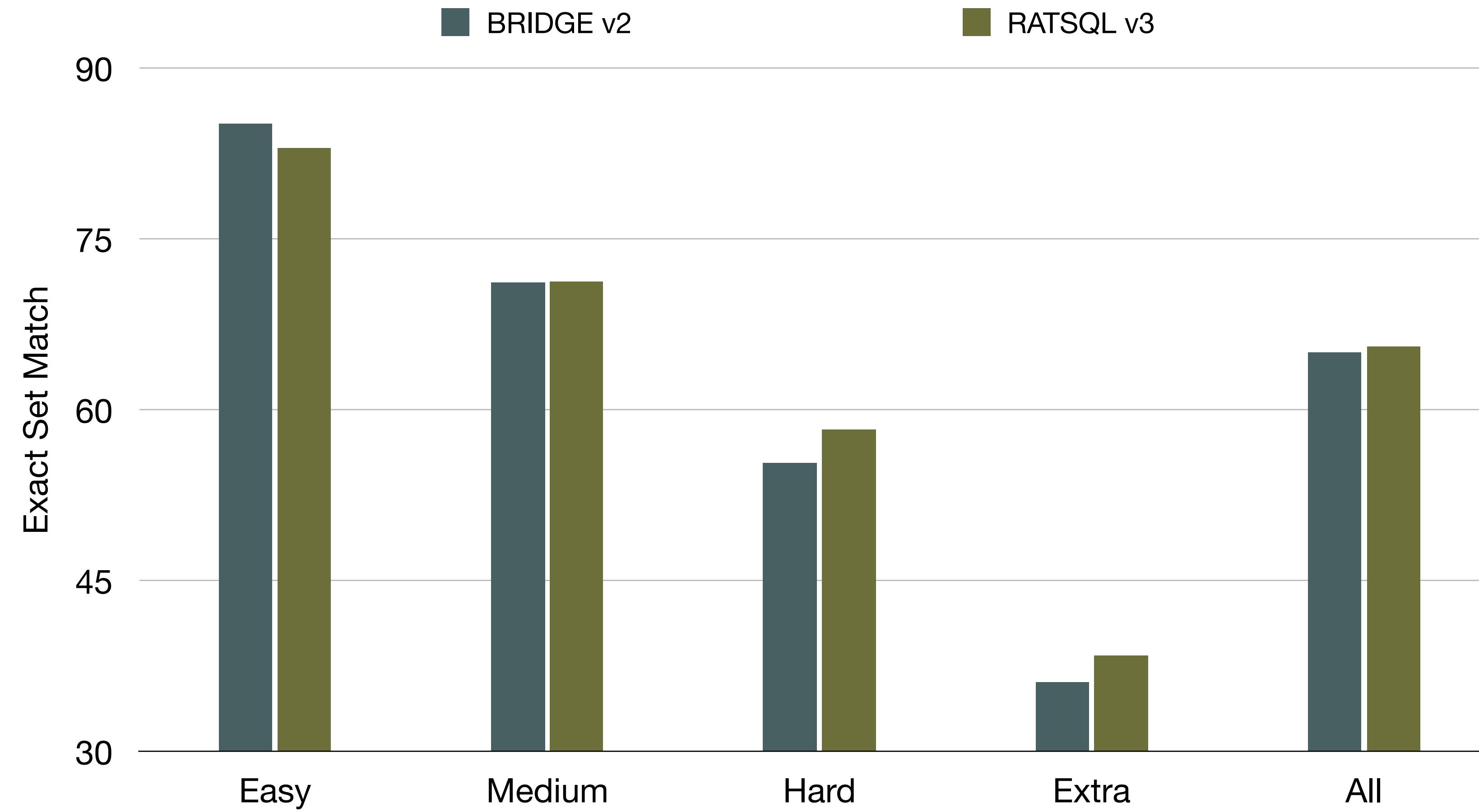


Figure 8. Performance of BRIDGE v2 compared to RATSQ v3 on the Spider Test set.

Performance Comparison by Difficulty Level

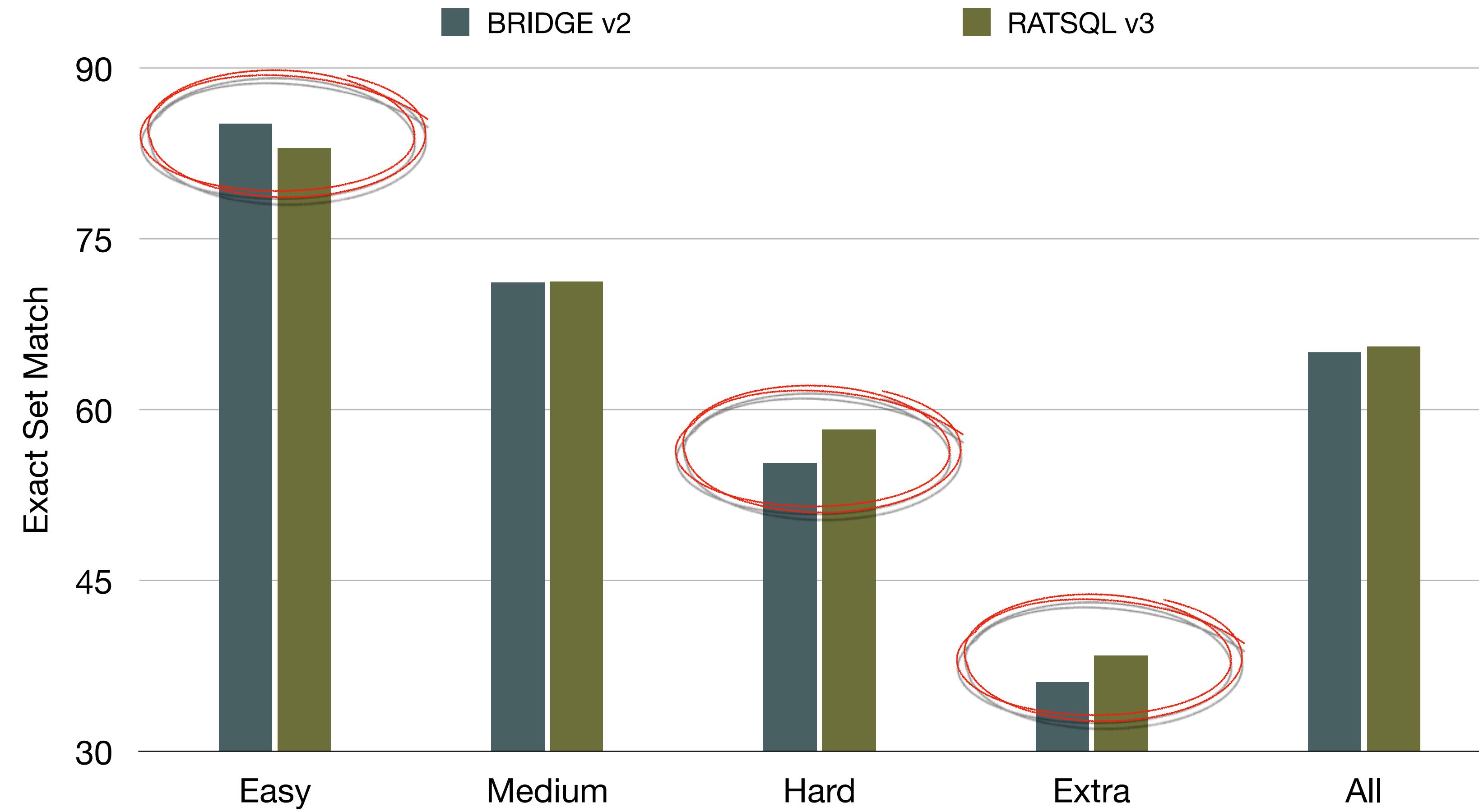


Figure 8. Performance of BRIDGE v2 compared to RATSQ v3 on the Spider Test set.

Performance on WikiSQL Leaderboard

Figure 9. Comparison to other top text-to-SQL models on the WikiSQL leaderboard (Jan 31, 2020). ♠ denotes approaches that use table content during training. EG refers to “execution guided decoding”.

Model	Dev		Test	
	EM	EX	EM	EX
SQLova (Hwang et al., 2019)	81.6	87.2	80.7	86.2
X-SQL (He et al., 2019b)	83.8	89.5	83.3	88.7
IE-SQL (Ma et al., 2020)	84.6	88.7	84.6	88.8
NL2SQL ♠ (Guo and Gao, 2019)	84.3	90.3	83.7	89.2
HydraNet (Lyu et al., 2020)	83.6	89.1	83.8	89.2
BRIDGE_L ♠	86.2	91.7	85.7	91.1
SQLova+EG (Hwang et al., 2019)	84.2	90.2	83.6	89.6
NL2SQL+EG ♠ (Guo and Gao, 2019)	85.4	91.1	84.5	90.1
X-SQL+EG (He et al., 2019b)	86.2	92.3	86.0	91.8
BRIDGE _L +EG ♠	86.8	92.6	86.3	91.9
HydraNet+EG (Lyu et al., 2020)	86.6	92.4	86.5	92.2
IE-SQL+EG (Ma et al., 2020)	87.9	92.6	87.8	92.5

Performance on WikiSQL Leaderboard

Figure 9. Comparison to other top text-to-SQL models on the WikiSQL leaderboard (Jan 31, 2020). ♠ denotes approaches that use table content during training. EG refers to “execution guided decoding”.

Model	Dev		Test	
	EM	EX	EM	EX
SQLova (Hwang et al., 2019)	81.6	87.2	80.7	86.2
X-SQL (He et al., 2019b)	83.8	89.5	83.3	88.7
IE-SQL (Ma et al., 2020)	84.6	88.7	84.6	88.8
NL2SQL ♠ (Guo and Gao, 2019)	84.3	90.3	83.7	89.2
HydraNet (Lyu et al., 2020)	83.6	89.1	83.8	89.2
BRIDGE _L ♠	86.2	91.7	85.7	91.1
SQLova+EG (Hwang et al., 2019)	84.2	90.2	83.6	89.6
NL2SQL+EG ♠ (Guo and Gao, 2019)	85.4	91.1	84.5	90.1
X-SQL+EG (He et al., 2019b)	86.2	92.3	86.0	91.8
BRIDGE _L +EG ♠	86.8	92.6	86.3	91.9
HydraNet+EG (Lyu et al., 2020)	86.6	92.4	86.5	92.2
IE-SQL+EG (Ma et al., 2020)	87.9	92.6	87.8	92.5

Best model without
execution guided
decoding

Performance on WikiSQL Leaderboard

Figure 9. Comparison to other top text-to-SQL models on the WikiSQL leaderboard (Jan 31, 2020). ♠ denotes approaches that use table content during training. EG refers to “execution guided decoding”.

Model	Dev		Test	
	EM	EX	EM	EX
SQLova (Hwang et al., 2019)	81.6	87.2	80.7	86.2
X-SQL (He et al., 2019b)	83.8	89.5	83.3	88.7
IE-SQL (Ma et al., 2020)	84.6	88.7	84.6	88.8
NL2SQL ♠ (Guo and Gao, 2019)	84.3	90.3	83.7	89.2
HydraNet (Lyu et al., 2020)	83.6	89.1	83.8	89.2
BRIDGE_L ♠	86.2	91.7	85.7	91.1
SQLova+EG (Hwang et al., 2019)	84.2	90.2	83.6	89.6
NL2SQL+EG ♠ (Guo and Gao, 2019)	85.4	91.1	84.5	90.1
X-SQL+EG (He et al., 2019b)	86.2	92.3	86.0	91.8
BRIDGE _L +EG ♠	86.8	92.6	86.3	91.9
HydraNet+EG (Lyu et al., 2020)	86.6	92.4	86.5	92.2
IE-SQL+EG (Ma et al., 2020)	87.9	92.6	87.8	92.5

Model	w/o EG		w/ EG	
	EM	EX	EM	EX
BRIDGE _L	86.2	91.7	86.8	92.6
-anchor text	84.2	90.0	85.2	91.3
-bridging	82.6	88.5	84.5	90.8

Best model without execution guided decoding

Performance on WikiSQL Leaderboard

Figure 9. Comparison to other top text-to-SQL models on the WikiSQL leaderboard (Jan 31, 2020). ♠ denotes approaches that use table content during training. EG refers to “execution guided decoding”.

Model	Dev		Test	
	EM	EX	EM	EX
SQLova (Hwang et al., 2019)	81.6	87.2	80.7	86.2
X-SQL (He et al., 2019b)	83.8	89.5	83.3	88.7
IE-SQL (Ma et al., 2020)	84.6	88.7	84.6	88.8
NL2SQL ♠ (Guo and Gao, 2019)	84.3	90.3	83.7	89.2
HydraNet (Lyu et al., 2020)	83.6	89.1	83.8	89.2
BRIDGE_L ♠	86.2	91.7	85.7	91.1
SQLova+EG (Hwang et al., 2019)	84.2	90.2	83.6	89.6
NL2SQL+EG ♠ (Guo and Gao, 2019)	85.4	91.1	84.5	90.1
X-SQL+EG (He et al., 2019b)	86.2	92.3	86.0	91.8
BRIDGE _L +EG ♠	86.8	92.6	86.3	91.9
HydraNet+EG (Lyu et al., 2020)	86.6	92.4	86.5	92.2
IE-SQL+EG (Ma et al., 2020)	87.9	92.6	87.8	92.5

Top-3 model using
execution guided
decoding

Cross-Database Performance

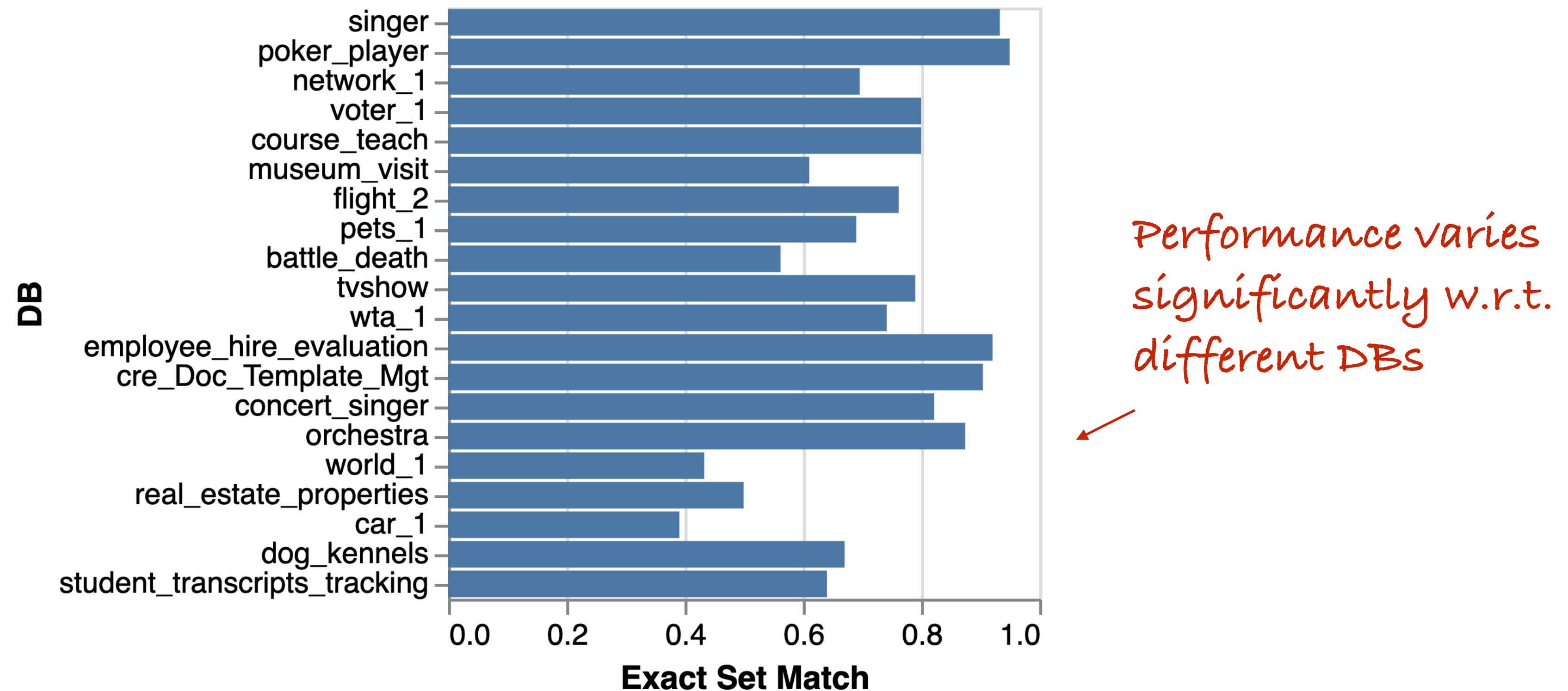


Figure 10. Performance of BRIDGE on each database on the Spider dev set.

Error Analysis

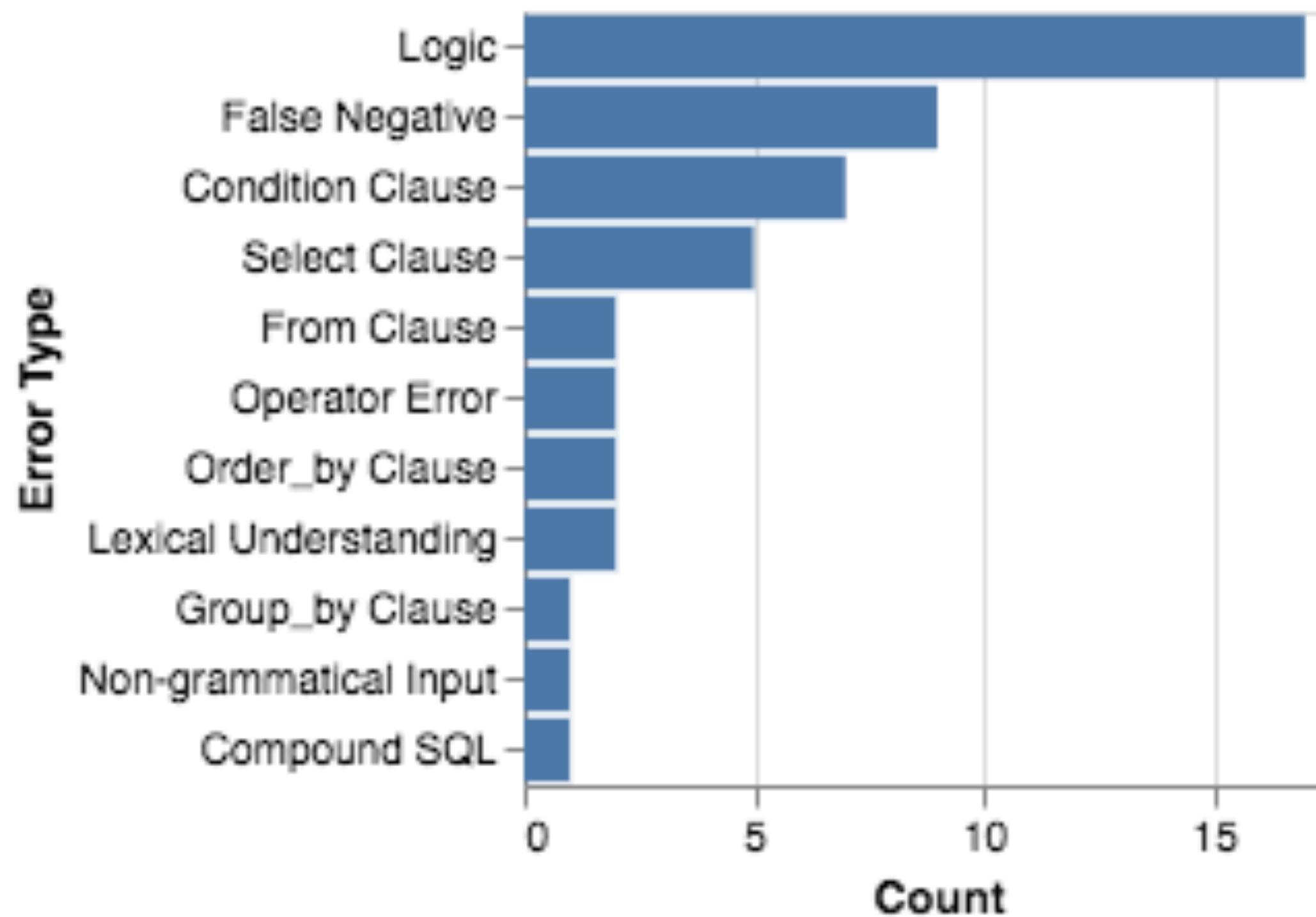


Figure 11. Manual error categorization for 50 wrong predictions on the Spider dev set.

Error Analysis

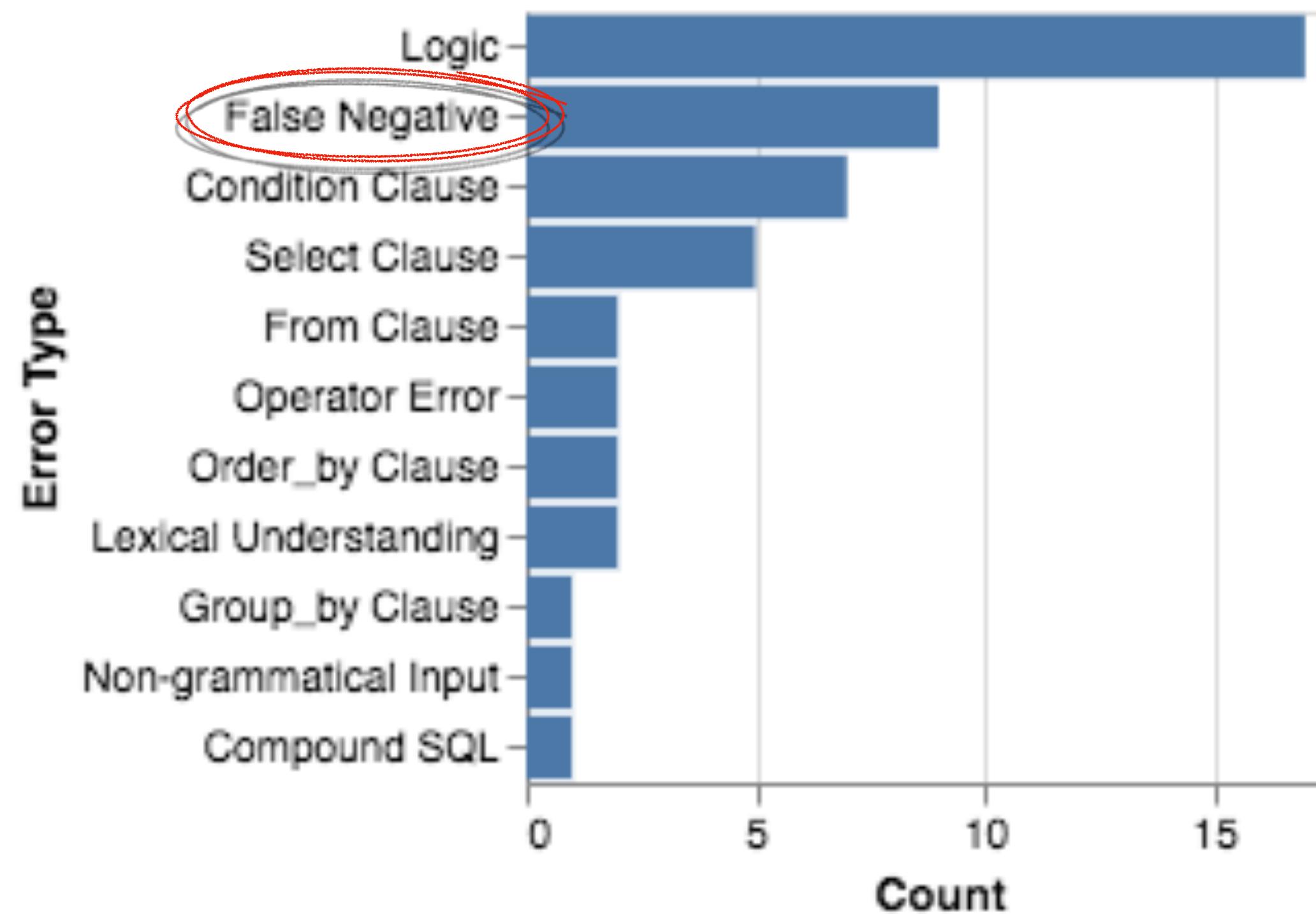


Figure 11. Manual error categorization for 50 wrong predictions on the Spider dev set.

Error Analysis

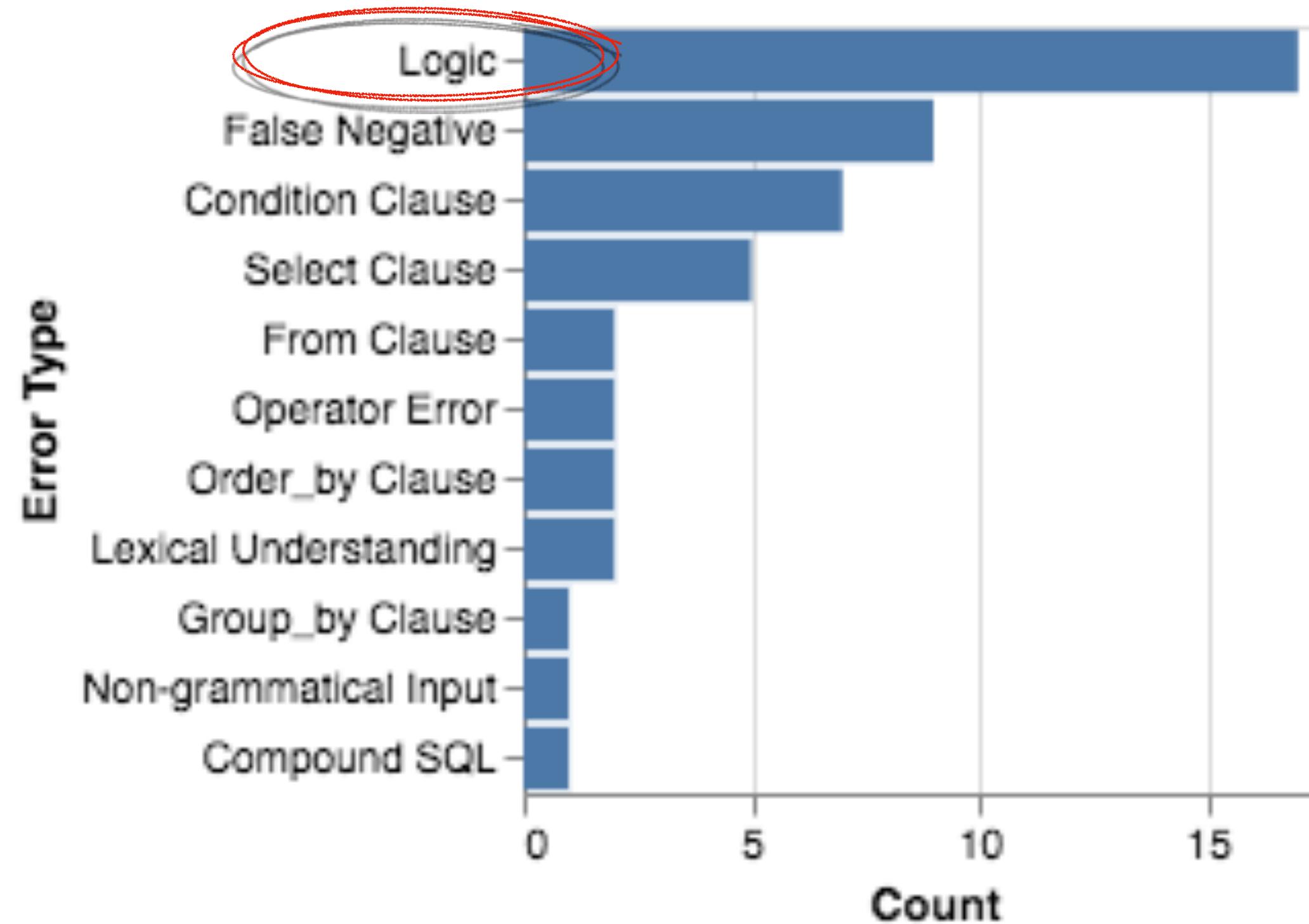


Figure 11. Manual error categorization for 50 wrong predictions on the Spider dev set.

Qualitative Examples

Error Category I - Logic

-
-  *Find the number of concerts happened in the stadium with the highest capacity.*
-  **X** SELECT COUNT(*) FROM stadium JOIN concert ON stadium.Stadium_ID = concert.Stadium_ID ORDER BY stadium.Capacity DESC LIMIT 1
-  **✓** SELECT COUNT(*) FROM concert WHERE stadium_id = (SELECT stadium_id FROM stadium ORDER BY capacity DESC LIMIT 1)
-  *Show the names of all of the high schooler Kyle's friends. **network_1***
- X** SELECT Highschooler.name FROM Friend JOIN Highschooler ON Friend.friend_id = Highschooler.ID WHERE Highschooler.name = "Kyle"
- ✓** SELECT T3.name FROM Friend AS T1 JOIN Highschooler AS T2 ON T1.student_id = T2.id JOIN Highschooler AS T3 ON T1.friend_id = T3.id WHERE T2.name = "Kyle"
-

Qualitative Examples

Error Category II - Lexical Understanding

Lexical Understanding	
	<p>👤 <i>Count the number of countries for which Spanish is the <u>predominantly spoken language</u>.</i></p> <p>world_1</p> <p>✗ SELECT COUNT(*) FROM countrylanguage WHERE countrylanguage.Language = "Spanish"</p> <p>✓ SELECT COUNT(*), MAX(Percentage) FROM countrylanguage WHERE LANGUAGE = "Spanish" GROUP BY CountryCode</p>
	<p>👤 <i>What are the full names of all <u>left handed</u> players, in order of birth date?</i> WTA_1</p> <p>✗ SELECT first_name, last_name FROM players ORDER BY birth_date</p> <p>✓ SELECT first_name, last_name FROM players <u>WHERE hand = 'L'</u> ORDER BY birth_date</p>

Qualitative Examples

Error Category III - Robustness

Robustness



What is the model of the car with the smallest amount of horsepower? **car_1**



`SELECT cars_data.Horsepower FROM cars_data ORDER BY cars_data.Horsepower LIMIT 1`



`SELECT T1.Model FROM CAR_NAMES AS T1 JOIN CARS_DATA AS T2 ON T1.MakeId = T2.Id ORDER BY T2.horsepower ASC LIMIT 1`



What is the total population and average area of countries in the continent of North America whose area is bigger than 3000? **concert_singer**



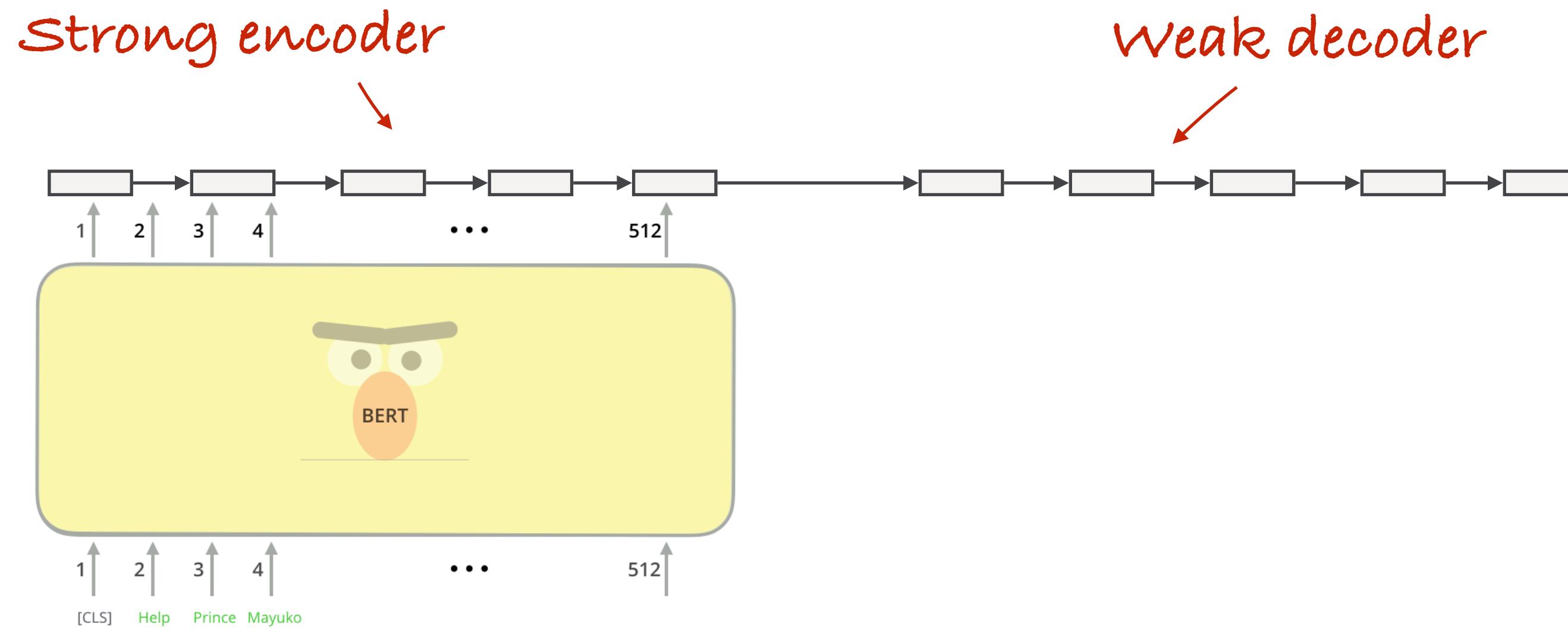
`SELECT SUM(country.Population), AVG(country.Population) FROM country WHERE country.Continent = "North America" AND country.SurfaceArea > 3000`



`SELECT SUM(country.population), AVG(country.surfacearea) FROM country WHERE country.Continent = "north america" and country.SurfaceArea > 3000`

Discussion

- I. BRIDGE is a sequence-to-sequence framework for jointly encoding **text**, **DB schema** and relevant **DB cells**, and output **SQL queries**.

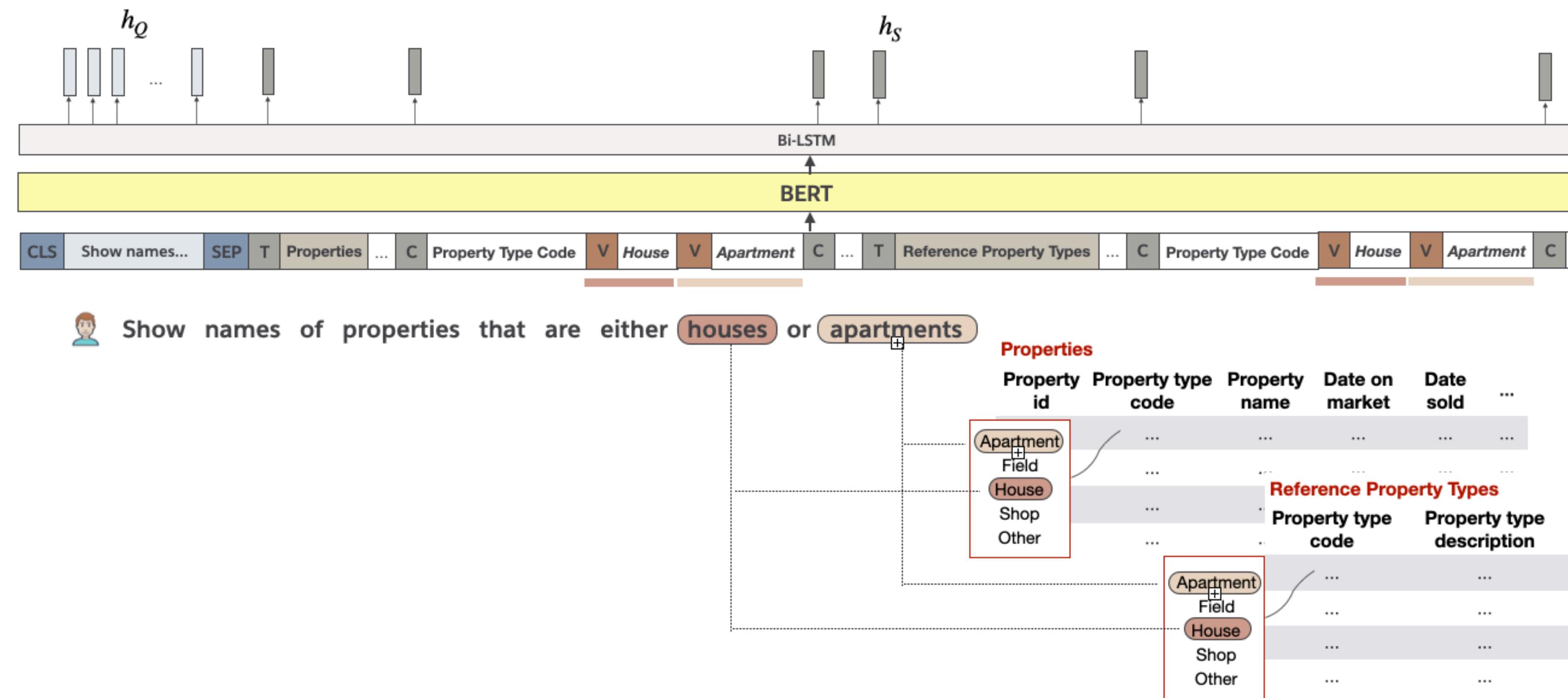


LM Alternatives:

- BART (Lewis et al. 2020)
 - T5 (Raffel et al. 2020)
 - GPT-2 (Radford et al. 2019)
 - GPT-3 (Brown et al. 2020)
- ...

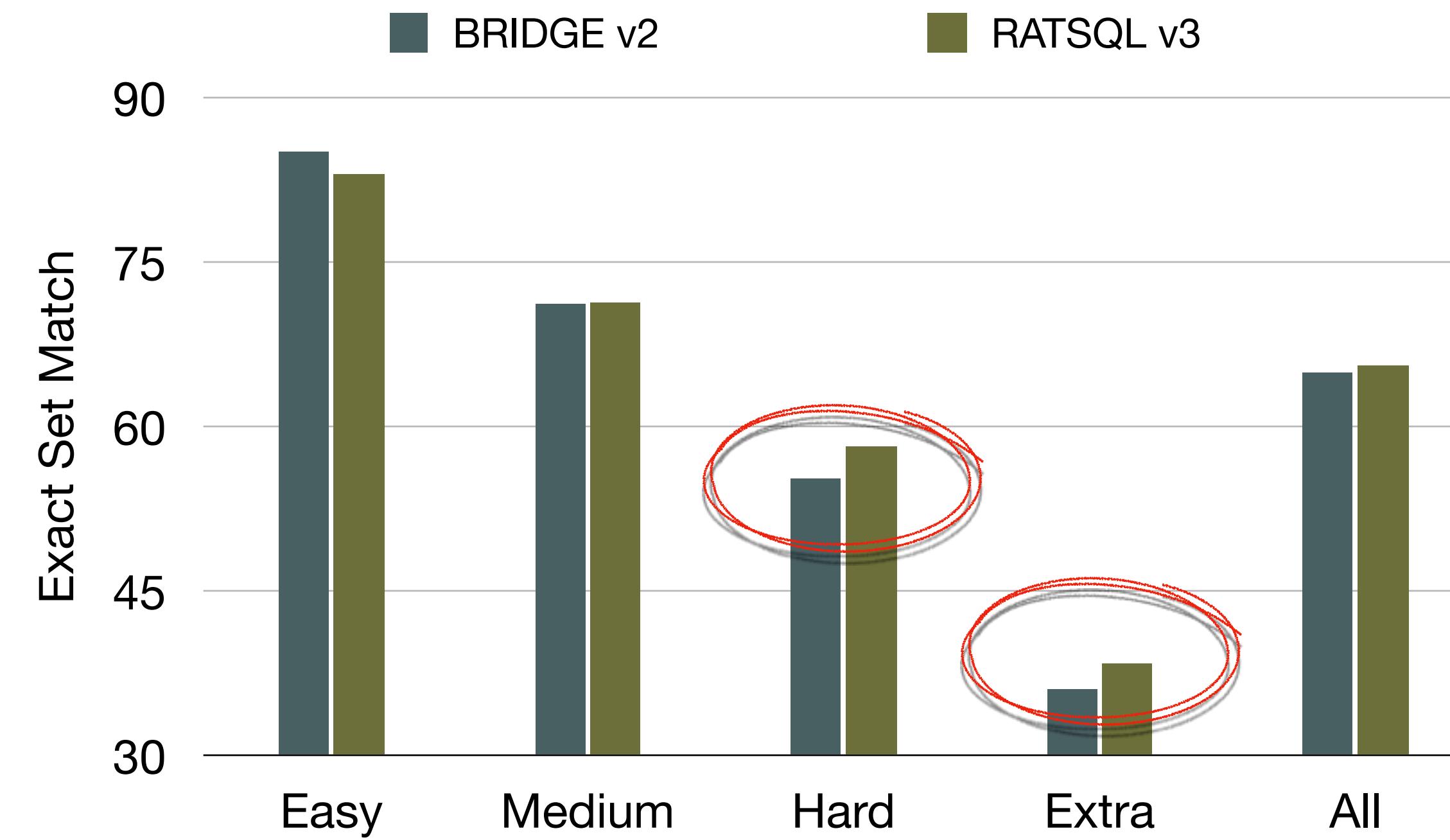
Discussion

II. Learning to recognize relevant cells (addressing acronyms and other lexical variations).



Discussion

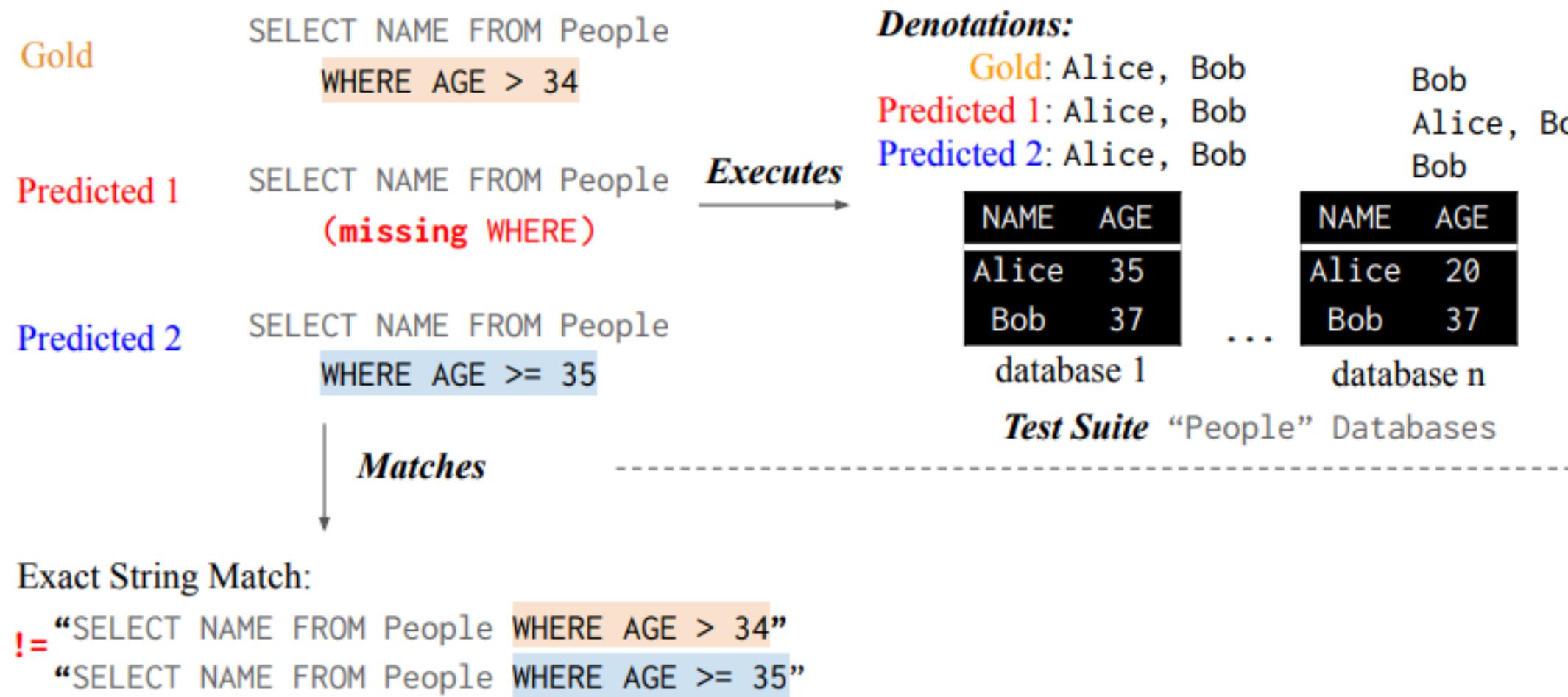
III. Compositional inductive biases (CIBs) demonstrates significant benefit when synthesize hard SQL queries. Previous work have shown that CIBs are effective for improving model's compositional generalization and overcome data sparsity.



Discussion

IV. Future work include more comprehensive model evaluation and benchmarking.

Query: "Who is above 34 years old?"



Distilled Test Suites by (Zhong et al. 2020)

Discussion

V. More future directions

- Train with execution feedback
- Overcome data sparsity
- Interpretability and Explainability
- Process context and pragmatics
- Question answering over DBs, documents, and other modality of information
- ...

Summary

- We introduce BRIDGE, a powerful sequence-to-sequence framework for cross-domain text-to-SQL semantic parsing
- BRIDGE uses pertained transformer LMs to capture language grounding of the input question, the DB schema, and the DB content

Co-authors:



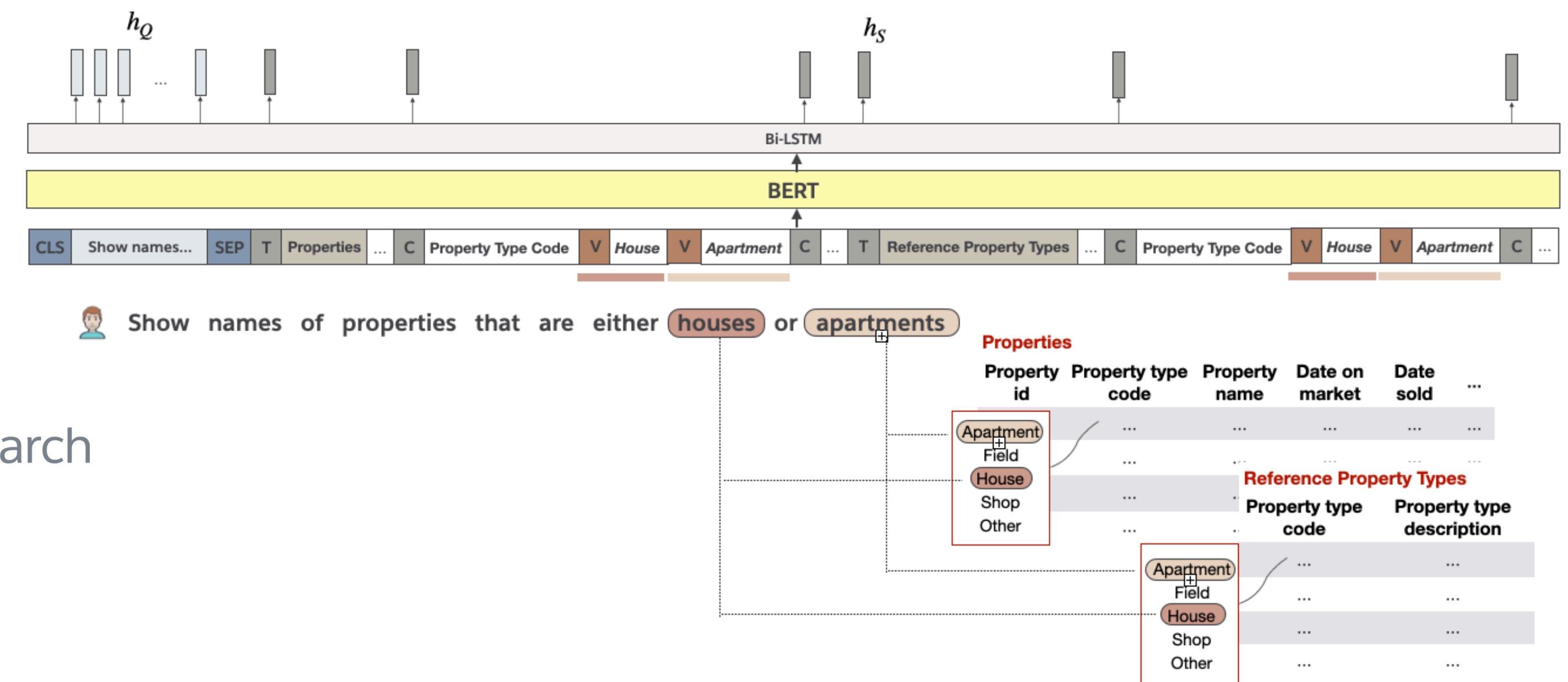
Richard Socher

You.com



Caiming Xiong

Salesforce AI
Research



Visualize Attention after Fine-tuning

