Data Democratisation with Deep Learning: The Anatomy of a Natural Language Interface

WSDM 2023 Tutorial

George Katsogiannis-Meimarakis (katso@athenarc.gr)
Mike Xydas (mxydas@athenarc.gr)
Georgia Koutrika (georgia@athenarc.gr)

DARE Lab

ATHENA

Presenters

George Katsogiannis

- **Research Assistant** at Athena Research Center, Greece
  - Text-to-SQL
  - Data Democratisation
  - INODE & FAIRCORE Projects
- MSc Student - Data Science and Information Technologies
  - Artificial Intelligence and Big Data specialisation

Mike Xydas

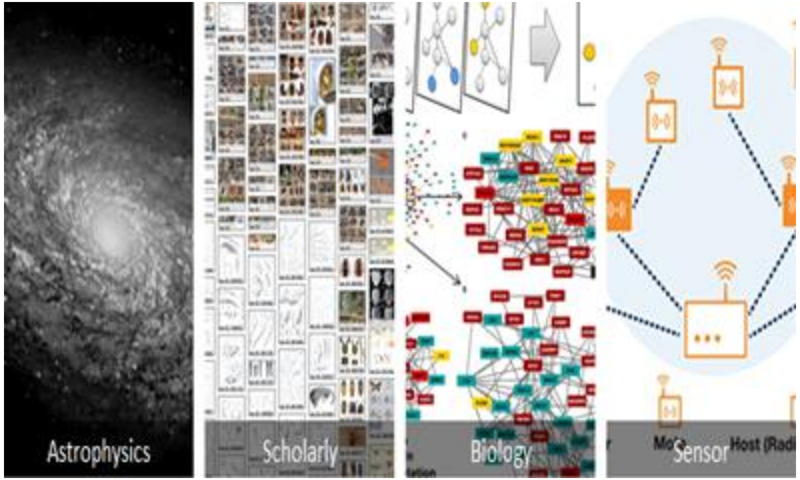
- **Research Assistant** at Athena Research Center, Greece
  - Data-to-Text
  - Data Democratisation
  - INODE & EOSCF Projects
- MSc Student - Data Science and Information Technologies
  - Artificial Intelligence and Big Data specialisation

Georgia Koutrika

- **Research Director** at Athena Research Center, Greece
- Research interests:
  - data exploration, including natural language interfaces, and recommendation systems
  - big data analytics
  - large-scale information extraction, entity resolution and information integration

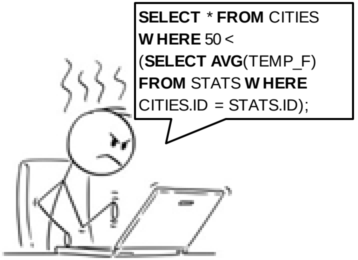
Why Natural Language Interfaces for Databases?

- The imminent **age of information** has made data an indispensable part of all human activities
- Many different **data sets are being generated** by users, systems and sensors
- Databases can benefit **many types of users** looking for insights, patterns, information, etc.
- However, not all users have **equal access to data**



Why Natural Language Interfaces for Databases?

- Data **volume and complexity** make it difficult to query data
- Database query interfaces are notoriously **user-UNFRIENDLY**



To access a database, the user must be familiar with SQL

What is **data democratisation**?

- **Empower everyone** to access, use, understand and derive value from data
- Lift the **technical barriers** that impede access to data and **eliminate dependency** to IT experts
- Design tools that are aimed for the **casual user**
- An organization-wide cultural stance

Why Natural Language?

To satisfy the needs of casual users of databases,
we must break through the barriers that presently prevent
these users from freely **employing their native languages**

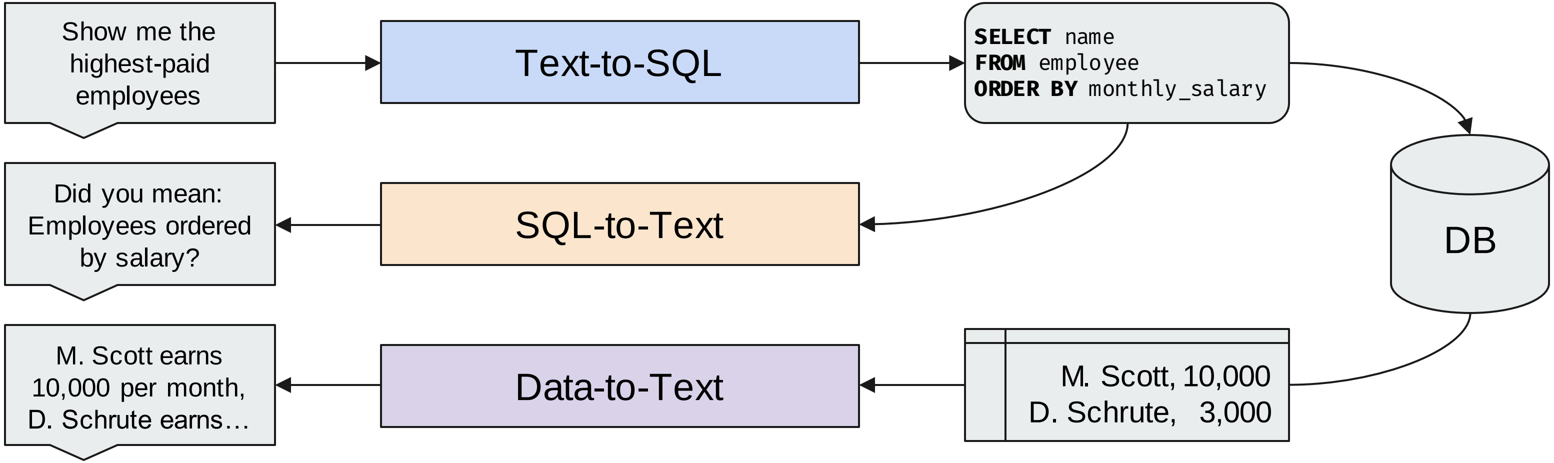
Which cities have year-round average temperature above 50 degrees?

Athens and Singapore have an average yearly temperature higher than 50°F

Ted Codd (circa: 1974)

Interacting with natural language can open up data access to everyone

What is a Natural Language Interface for Databases?



Tutorial Outline

Part 1

Text-to-SQL

1. The Text-to-SQL problem
2. Benchmarks
3. A Taxonomy for Deep Learning Text-to-SQL Systems
4. Key Systems
5. Research Challenges

Part 2

SQL-to-Text

1. The SQL-to-Text problem
2. Challenges
3. Key Systems
4. Research Challenges

30 min.

Part 3

Data-to-Text

1. What is Data-to-Text
2. Subfields of Data-to-Text
3. Table-to-Text
4. Graph-to-Text
5. Evaluation
6. Research Challenges

Part 4

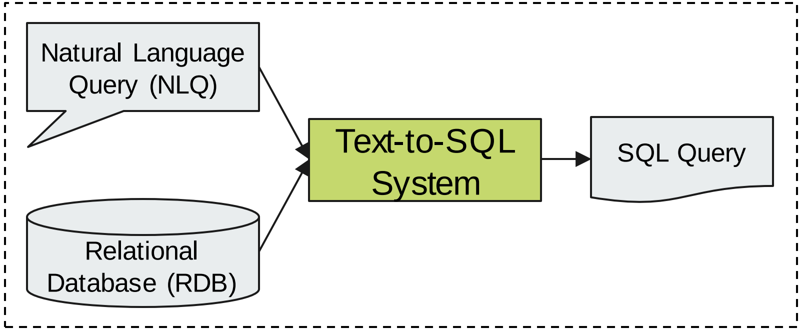
Bringing it all together

1. What do we mean?
2. Why is it not trivial?
3. Challenges
4. Demo

The Text-to-SQL Problem

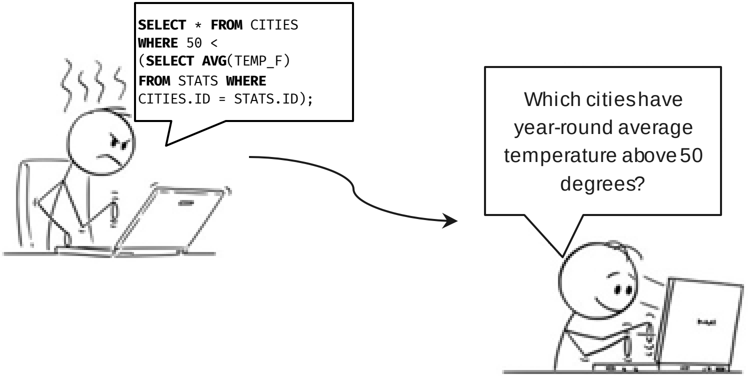
The Text-to-SQL Problem

"Given a Natural Language Query (NLQ) on a Relational Database (RDB) with a specific schema, produce a SQL query equivalent in meaning, which is valid for the said RDB and that when executed will return results that match the user's intent."



The Text-to-SQL Problem

- The text-to-SQL problem has long been a holy grail for the DB community
- It would allow users to query DBs without any technical skills
- There have been many efforts from the DB community during the past decades
- However this is a notoriously difficult problem



Users can query the DB with NL instead of SQL

Challenges: From the NL side

- Complexity of NL
  - Ambiguity
  - References - Schema Linking
  - Inferences
  - Vocabulary Gap
- User Mistakes
  - Spelling mistakes
  - Syntactical/Grammatical mistakes

City or person?

"Show information about Paris"

"model" refers to car.model OR engine.model ?

"President *(of the USA)* before Obama?"

"composer" vs "songwriter"

Grammys

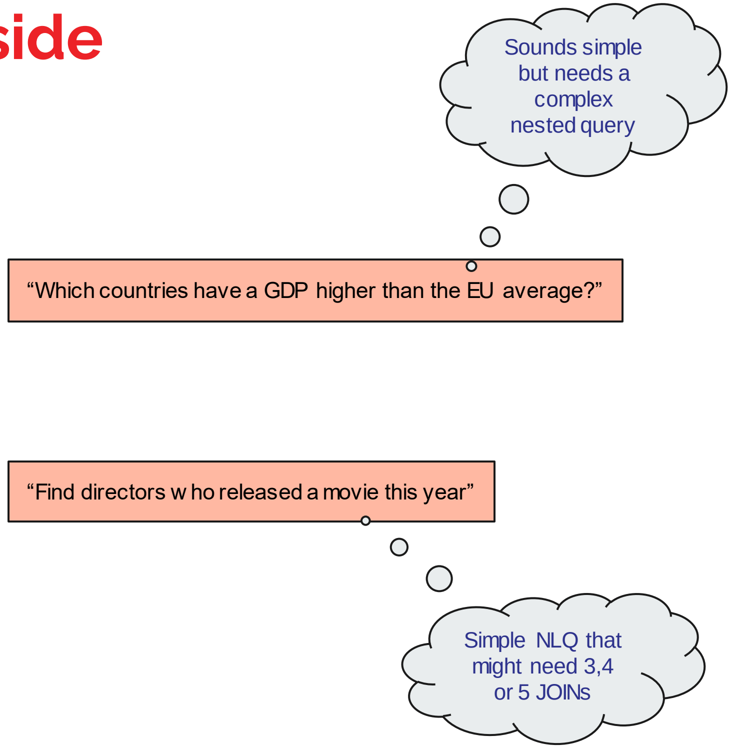
"Which singer won the most Grammies?"

"Show most actor played movies "

??

Challenges: From the SQL side

- Complex Syntax
  - SQL is a structured language with a strict grammar and limited expressivity
- Database Structure
  - The user’s data model may not match the data schema

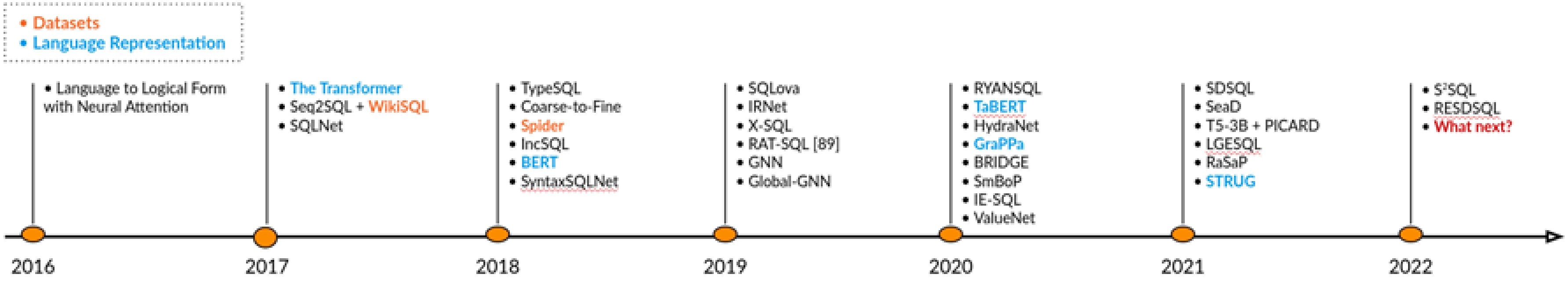


Text-to-SQL: Early Approaches

Syntactic parsers, ontology mappings, knowledge bases, information retrieval…

Why not use deep learning and treat it like a translation problem from NLQ to SQL?

- Keyword Systems [46, 47, 48]
  - Search engine-like functionality, where NLQs contain just keywords
  - e.g., *"drama movies"*
- Enhanced Keyword systems [49, 50]
  - Queries with aggregate functions, comparison operators, and keywords that map to database metadata
  - Syntactic constraints on their input to make sure they can parse the user query
  - e.g., *"count movies actress "Priyanka Chopra""*
- Natural language systems [51, 52]
  - Allow queries in natural language
  - e.g., *"What is the number of movies of "Priyanka Chopra""*



A brief timeline of deep learning text-to-SQL research

Available Benchmarks

Text-to-SQL Benchmarks

Several pain points of early evaluation:

- ✗ **No common datasets**
  - System evaluations have used different datasets of varying size and complexity.
- ✗ **Small or proprietary datasets**
  - e.g., TPC-H (100MB) and DBLP (56MB)
- ✗ **No standard, small query sets**
  - Different test queries, often not available to reproduce the experiments.
- ✗ **Incomparable effectiveness evaluations**
  - none, user study, manual evaluation, comparison to gold standard queries

Year	Dataset	Examples	Databases
1994	ATIS	275	1
1996	GeoQuery	525	1
2003	Restaurants	39	1
2014	Academic	179	1
2017	IMDb	111	1
	Yelp	68	1
	Scholar	396	1
	WikiSQL	**80,654**	**24,241**
2018	Advising	281	1
	Spider	**10,181**	**200**
2020	MIMICSQL	10,000	1
	SQUALL	11,276	1,670
	FIBEN	300	1
2021	Spider-Syn	8,034	160
	Spider-DK	535	?
	KaggleDBQA	272	8
	SEDE	12,023	1

Two large cross-domain benchmarks revolutionised text-to-SQL research, opening the door to machine learning

WikiSQL

- Large crowd-sourced dataset for developing NL interfaces for relational databases
  - 80K NL/SQL pairs over 25K tables
- NL questions on tables gathered from Wikipedia
  - Not entire databases!
  - The SQL queries that can be performed are quite simple
- Contains many mistakes
  - Research suggests that the upper bound has been reached
  - Human accuracy estimated at 88%

NLQ: What nationality is the player Muggsy Bogues?

```
SELECT nationality
WHERE player = muggsy bogues
```

Player	No.	Nationality	Position	Years in Toronto	School/ Team
Leandro Barbosa	20	Brazil	Guard	2010-2012	Tilibra
Muggsy Bogues	14	USA	Guard	1999-2001	Wake Forest
...	...	...	...	...	...

Spider

- Large-scale complex and cross-domain semantic parsing and text-to-SQL dataset
  - 10,181 questions
  - 5,693 complex SQL queries
  - 200 databases from 138 different domains
- Annotated by 11 Yale students
- Queries of varying complexity
  - Categories of difficulty
  - SQL elements such as JOIN, GROUP BY, UNION, INTERSECT, nested queries
- Better quality and complexity than WikiSQL

Easy

What is the number of cars with more than 4 cylinders?

```
SELECT COUNT(*)
FROM cars_data
WHERE cylinders > 4
```

Medium

For each stadium, how many concerts are there?

```
SELECT T2.name, COUNT(*)
FROM concert AS T1 JOIN stadium AS T2
ON T1.stadium_id = T2.stadium_id
GROUP BY T1.stadium_id
```

Hard

Which countries in Europe have at least 3 car manufacturers?

```
SELECT T1.country_name
FROM countries AS T1 JOIN continents
AS T2 ON T1.continent = T2.cont_id
JOIN car_makers AS T3 ON
T1.country_id = T3.country
WHERE T2.continent = 'Europe'
GROUP BY T1.country_name
HAVING COUNT(*) >= 3
```

Extra Hard

What is the average life expectancy in the countries where English is not the official language?

```
SELECT AVG(life_expectancy)
FROM country
WHERE name NOT IN
  (SELECT T1.name
   FROM country AS T1 JOIN
   country_language AS T2
   ON T1.code = T2.country_code
   WHERE T2.language = "English"
     AND T2.is_official = "T")
```

A Taxonomy of Text-to-SQL Deep Learning Systems

Discovery of possible mentions of database elements (tables, attributes and values) in the NLQ

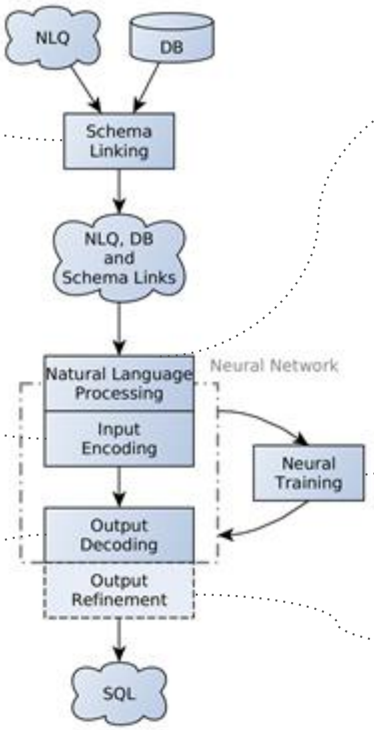
Creating an efficient numerical representation that can be read by the encoder

Selection of the encoder and structuring the inputs so that they can be accepted by the encoder

How to train the network for the best possible performance

Designing the structure of the predictions that the network will make

Additional techniques that can be added after training to improve performance



A Taxonomy of Text-to-SQL Deep Learning Systems

Taxonomy Overview of a Deep Learning Text-to-SQL system

Schema Linking

Finding connections between the NLQ and the DB

- Consider a human writing a SQL query based on a NL specification
- Important to find how elements of the NL appear in the DB
- Three main types of schema links:
    - Table links
    - Column links
    - Value links

NLQ:

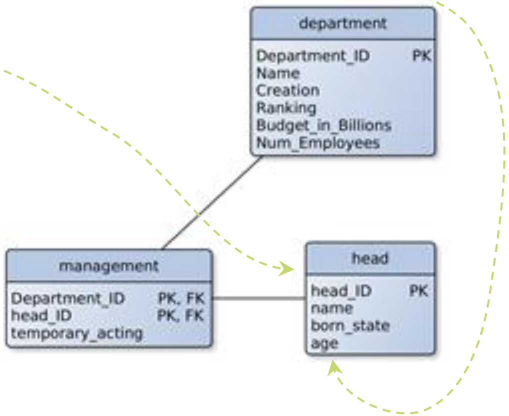
Table link: *head*

Value link: *age*

How many heads of the departments are older than 56 ?

SQL:

```
SELECT COUNT(*)
FROM head
WHERE age > 56
```



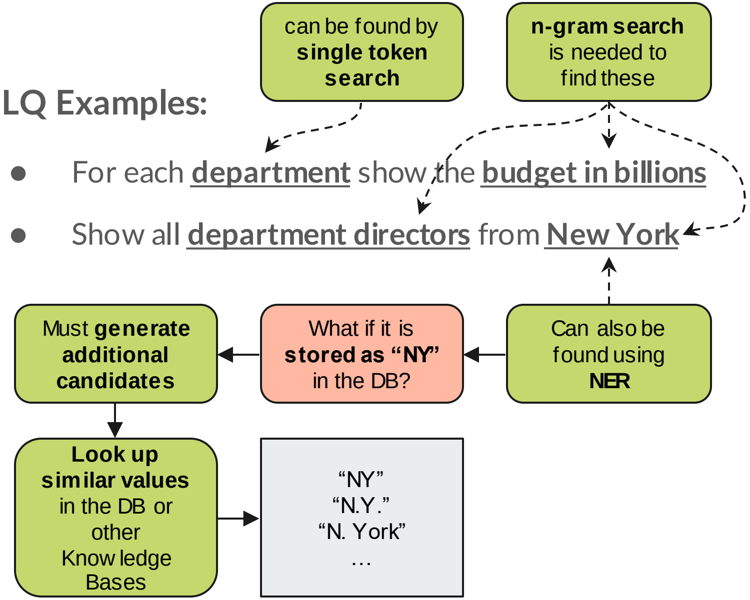
Schema Linking: Query Candidates

The three questions of schema linking:

- Which parts of the **NLQ** to consider?
    - Single Tokens
    - Multi-word candidates (n-grams)
    - Named Entities
    - Generate Additional Candidates
- Which parts of the **DB** to consider?
- How to decide on a **match**?

NLQ Examples:

- For each **department** show the **budget in billions**
- Show all **department directors** from **New York**



Schema Linking: Database Candidates

The three questions of schema linking:

- Which parts of the **NLQ** to consider?
- Which parts of the **DB** to consider?
  - Table and Column Names
  - Values via Lookup
  - Values via Knowledge Graphs
- How to decide on a **match**?

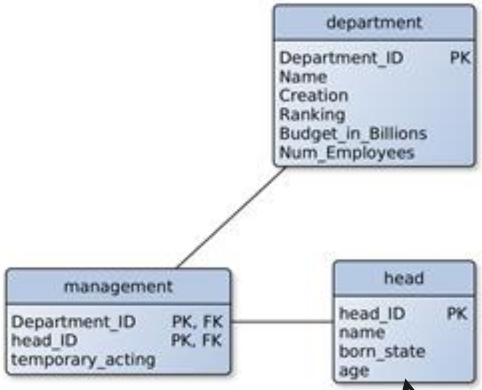
Need an efficient method due to large size of data

Database structures such as inverted indices

What if access to the data is not available?

We can search candidates such as “New York” in external KGs

ConceptNet informs us that “New York” is a **state**

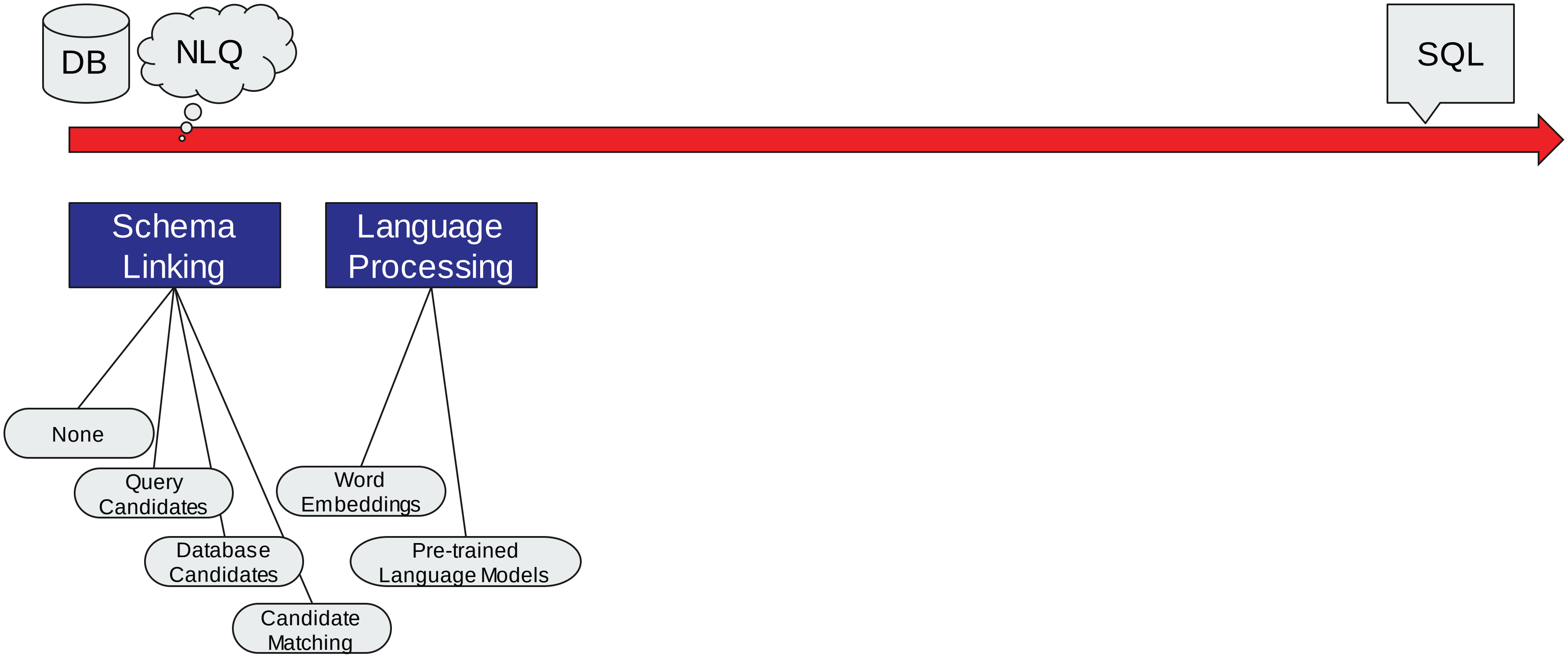


Schema Linking: Candidate Matching

The three questions of schema linking:

- Which parts of the **NLQ** to consider?
- Which parts of the **DB** to consider?
- How to decide on a **match**?
    - Exact and partial match
    - Fuzzy/Approximate String Matching
    - Learned Embeddings
    - Classifiers

Query Candidate	DB Candidate	Match Method
“department”	“department”	Exact Match
“budget”	“budget in billions”	Partial Match
“dept.”	department	Fuzzy Match
“department director”	“head”	Learned Embeddings
		Classifiers



Taxonomy Overview of a Deep Learning Text-to-SQL system

Natural Language Processing

How can we give natural language to a neural network?

- LSTM Neural Networks (1995) 🔗 [14]
- Word Embeddings
  - One-hot Embeddings
  - Word2Vec (2013) 🔗 [15]
  - GloVe (2014) 🔗 [16]

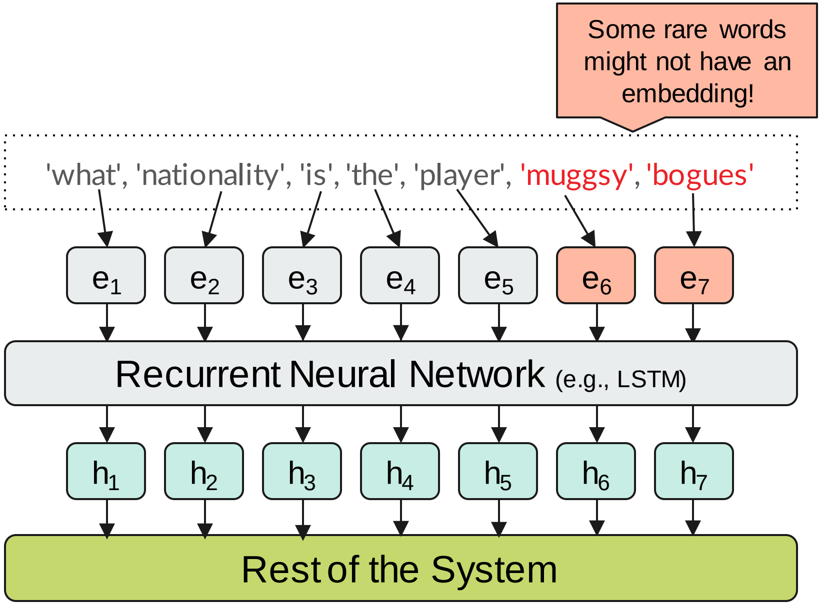
- The Transformer (2017) 🔗 [9]
- The rise of language models
  - BERT (2018) 🔗 [10]
  - RoBERTa (2019) 🔗 [11]
  - TaBERT (2020) 🔗 [12]
  - GraPPa (2020) 🔗 [13]
  - BART (2020) 🔗 [17]
  - T5 (2020) 🔗 [18]

encoder-only

encoder-decoder

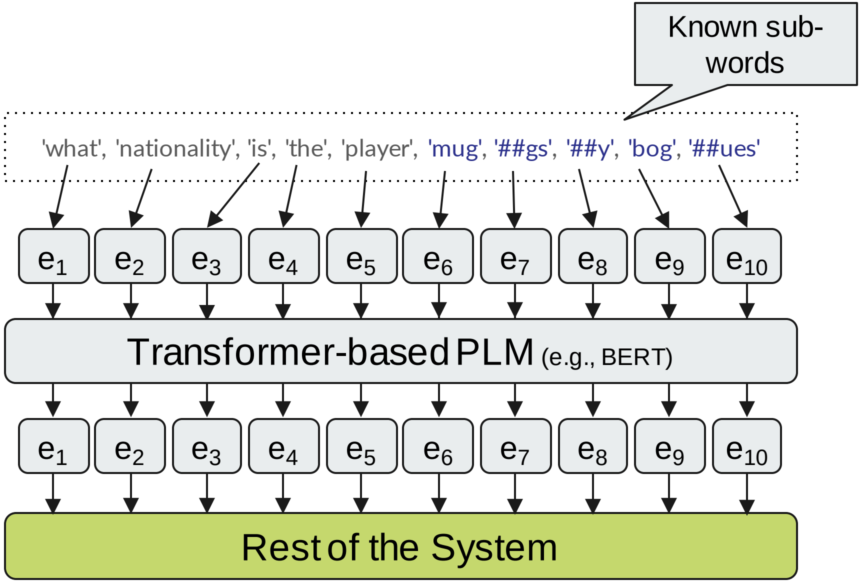
Using Word Embeddings

- Each word of the input is assigned to a pre-trained word embedding vector
  - **Out of vocabulary** problem
- The embedding sequence is then processed by a RNN to create a hidden representation
- Major drawbacks of RNNs:
  - Large **processing costs** for long sequences
  - Hard to make associations of words that are not near each other



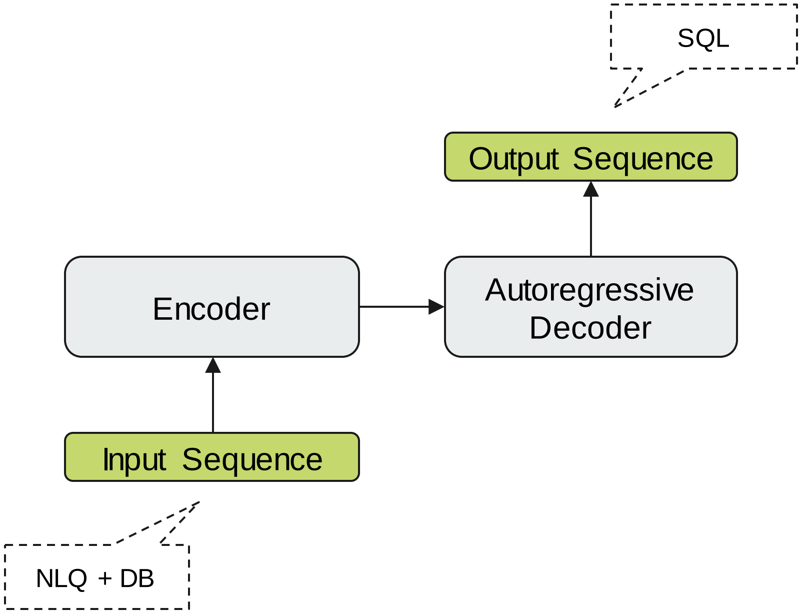
Using Encoder-only PLMs

- Transformer architecture speeds up processing, even for large inputs
- The vast amounts of NL data seen during pre-training is beneficial for performance
- Words are split to known sub-words, using wordpiece embeddings
  - Eliminates the out of vocabulary problem
- Greatly increases hardware requirements



Encoder-Decoder PLMs

- Another category of very powerful transformer-based pre-trained models
- Operate on a **sequence-to-sequence** (text-to-text) framework
- Limited design choices, but very good results (e.g., T5-3B + PICARD)

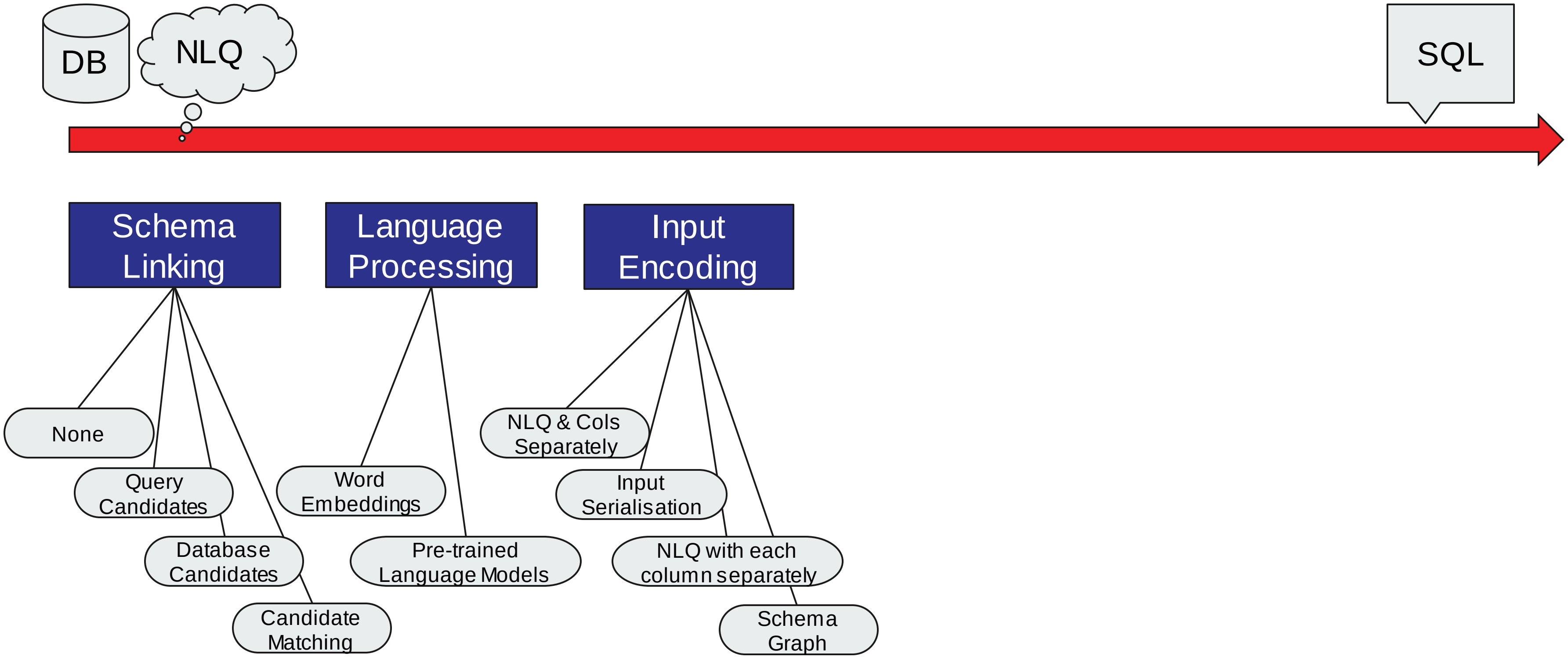


Task-specific PLMs: GraPPa

- Initialized by RoBERTa-Large
- Synthetic pre-training **data** is created from tabular datasets like:
  - Spider
  - WikiSQL
  - WikiTableQuestions
- Experiments show **better performance in text-to-SQL** when using GraPPa instead of RoBERTa

Pre-training **tasks**:

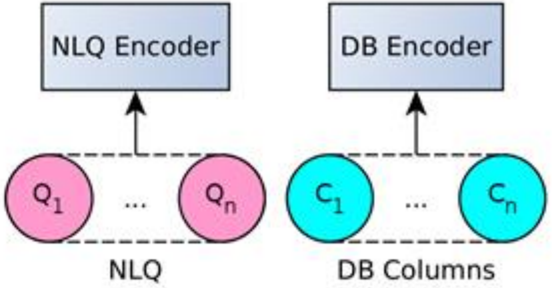
- Masked Language Modelling (MLM)
  - Input: NLQ/Table Description + Columns
  - The network must **predict the masked words** both in the NLQ and columns
- SQL Semantic Prediction (SSP)
  - Input: NLQ + Columns
  - The network must predict for each column, **if it appears in the SQL and its role** (e.g. SELECT, GROUP BY)



Taxonomy Overview of a Deep Learning Text-to-SQL system

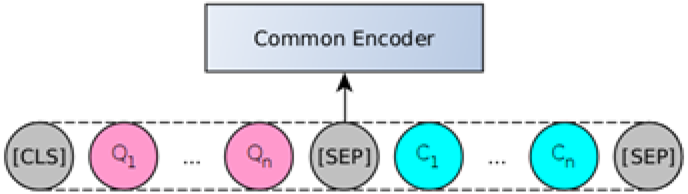
Input Encoding: Separate Encoding

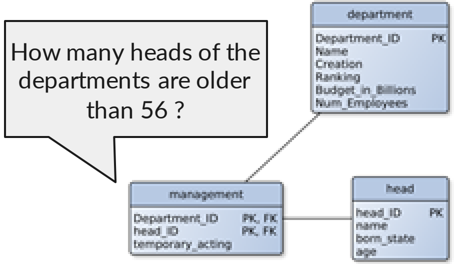
- Used by the first text-to-SQL systems (Seq2SQL, SQLNet) for WikiSQL
- The main reason is the **different format** of the NLQ and table columns
  - **NLQ:** Sequence of words
  - **Column names:** Sequence of sequences of words
- The two different inputs **must be combined** (attention, concatenation, sum, etc.)



Input Encoding: Serialisation

- Widely used by newer systems incorporating language models
- No need to combine different inputs
- The database schema is flattened into a sequence of words

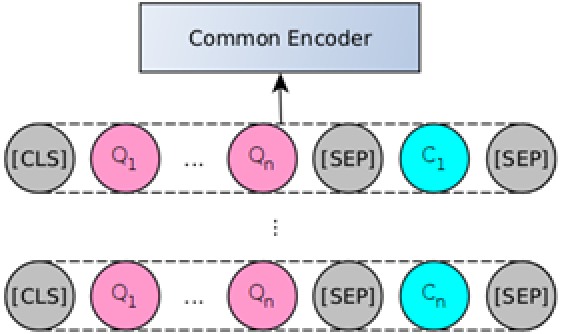




'How', 'many', 'heads', 'of', 'the', 'departments', 'are', 'older', 'than', '56, '?', [SEP], 'department', [SEP], 'name', [SEP], 'creation', [SEP], 'ranking', [SEP], 'budget_in_billions', [SEP], 'num_employes', [SEP], 'management', [SEP], 'department_id', [SEP], 'head_id', [SEP], 'temporary_acting', [SEP], 'head', [SEP], 'head_id', [SEP], 'name', [SEP], 'born_state', [SEP], 'age', [SEP]

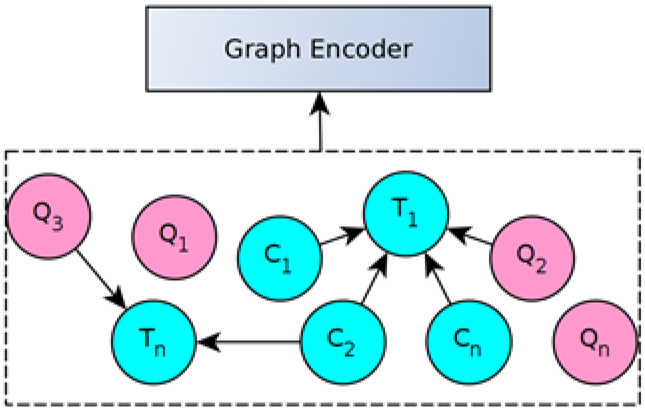
Input Encoding: NLQ with Each Column Separately

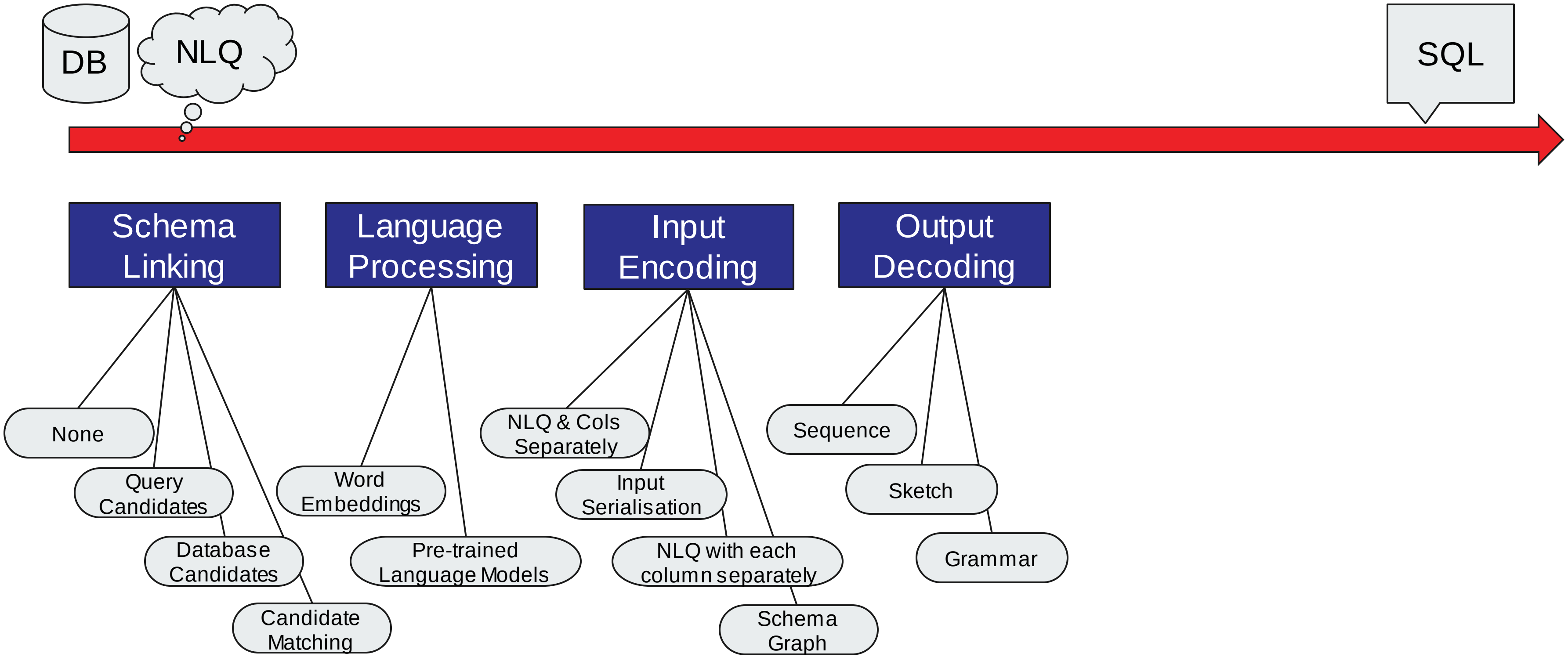
- A unique approach proposed by **HydraNet**
- The NLQ is **processed** with each column **separately**
- **Predictions** are made for each column **separately**
- Works very well on **WikiSQL**
- No similar approach for **Spider**



Input Encoding: Graph Encoding

- Using graphs allows the preservation of all the **schema relations**
  - Which columns belong to which table
  - Which columns are keys
  - Which tables are connected by foreign keys
- The **words of the NLQ** can be added to the graph based on schema links and similarity
- Much more **complex** neural design





Taxonomy Overview of a Deep Learning Text-to-SQL system

Output Decoding: Sequence-based

[19] Language to Logical Form with Neural Attention (2016)
[2] Seq2SQL (2017)
[20] BRIDGE (2020)
[21] T5-3B + PICARD (2021)

- We generate the SQL output as a simple text sequence
- Any sequence-to-sequence architecture is compatible
- The network must learn to generate SQL tokens, with the correct syntax!

Simplifies the text-to-SQL problem

More possibilities for errors

- Nothing prevents syntactical errors when predicting
- Usually avoided until recently
- Recent works show promising techniques that help avoid such errors

Output Decoding: Sketch-based

- [22] SQLNet (2017)
- [23] SQLova (2019)
- [24] HydraNet (2020)

- We have a sketch of the query with missing parts that need to be filled
- Sketch used by systems designed for WikiSQL

```
SELECT <AGG> <COLUMN>
(
     WHERE <COLUMN> <OP> <VALUE>
     ( AND <COLUMN> <OP> <VALUE> ) *
) ?
```

- Further simplifies the task of producing a SQL query into smaller sub-tasks
- Hard to extend for complex queries with additional clauses (e.g., GROUP BY, JOIN, nested queries, etc.)

Output Decoding: Grammar-based

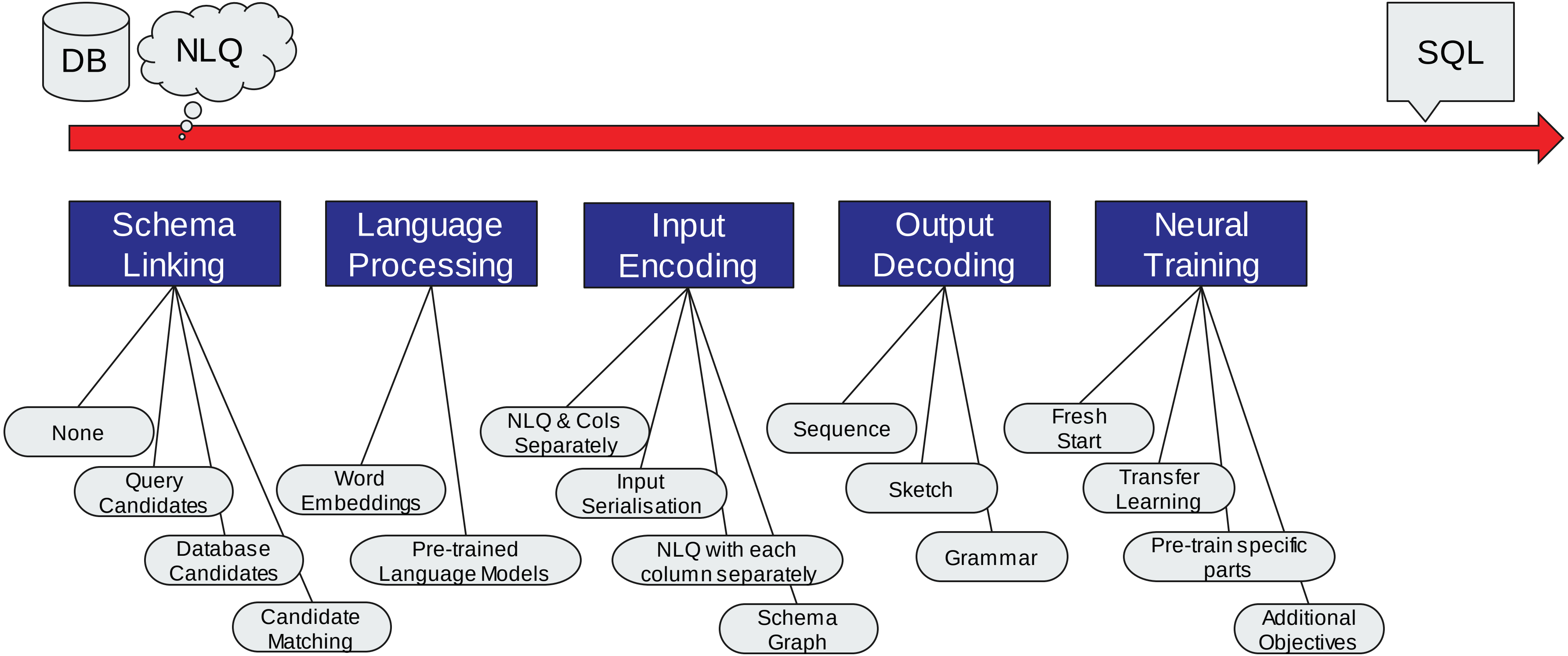
[25] IncSQL (2018)
[6] IRNet (2019)
[20] RAT-SQL (2020)

- Generate a sequence of rules instead of simple tokens
- Apply the rules sequentially to get a SQL query

Easier to avoid errors

Can cover more complex SQL queries

Needs more complex design



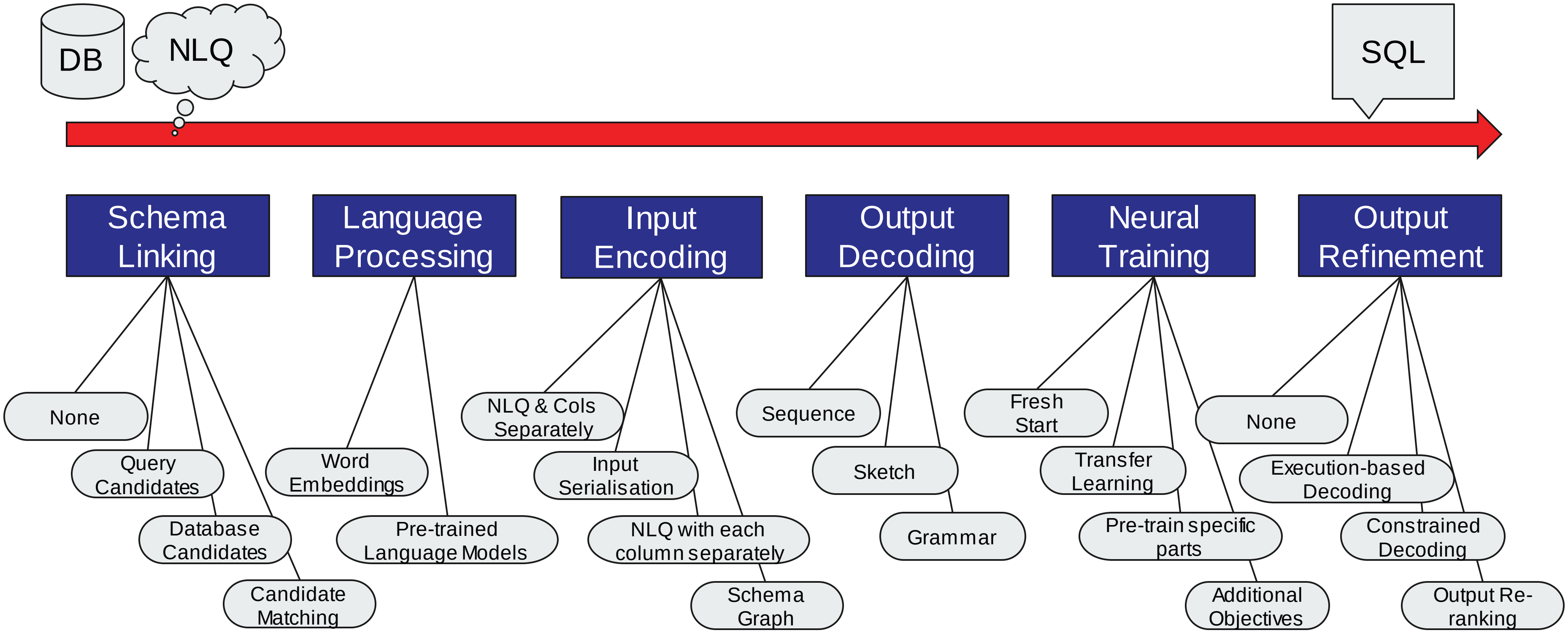
Taxonomy Overview of a Deep Learning Text-to-SQL system

Neural Training

Erosion: Delete parts of the DB schema and train the model to produce the correct SQL with the eroded schema

Shuffling: Randomly change the order of attributes and conditions in both the NLQ and SQL and train the network to re-order them correctly

1. **Fresh Start:** Train the network from scratch
2. **Transfer Learning:** First pre-train on a generic task, then fine-tune for text-to-SQL
   - The Computer Vision and NLP domains have proven its power
   - Has seen widespread use with the introduction of Transformer-based PLMs
3. **Additional Objectives:** Train for additional sub-tasks simultaneously with text-to-SQL
   - Training for additional tasks, related to the main problem, can boost performance
4. **Pre-train Specific Parts:** Maybe some components of the network can benefit by independent pre-training
   - GP proposes to pre-train the decoder, in order to better learn the output's grammar



Taxonomy Overview of a Deep Learning Text-to-SQL system

Output Refinement: Execution-Guided Decoding

- Sketch-based approaches greatly **reduce** the possibility of errors
- There are still a few possibilities
  - **Aggregation function mismatch** (e.g. AVG on string type)
  - **Condition type mismatch** (e.g. comparing a float type column with a string type value)

- Execution guided decoding helps the system **avoid** making such choices at **prediction time**
- By executing **partially complete** predicted SQL queries, the system can reject choices that create **execution errors** or **yield empty results**

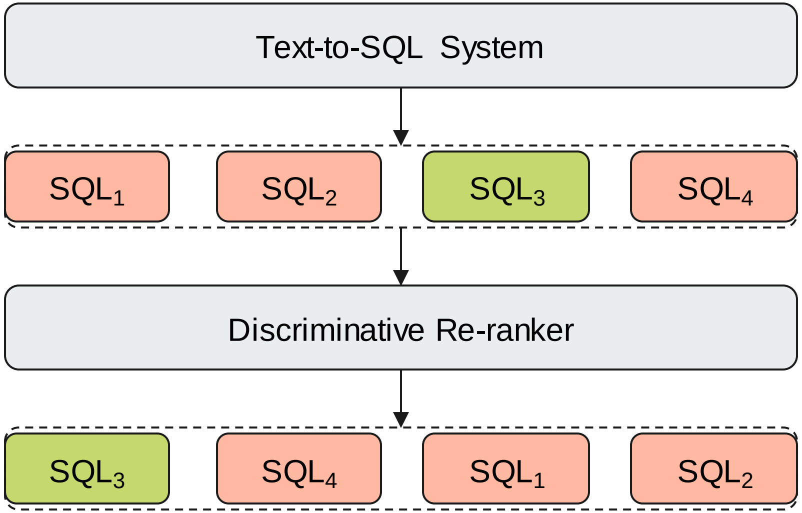
Output Refinement: Constrained Decoding

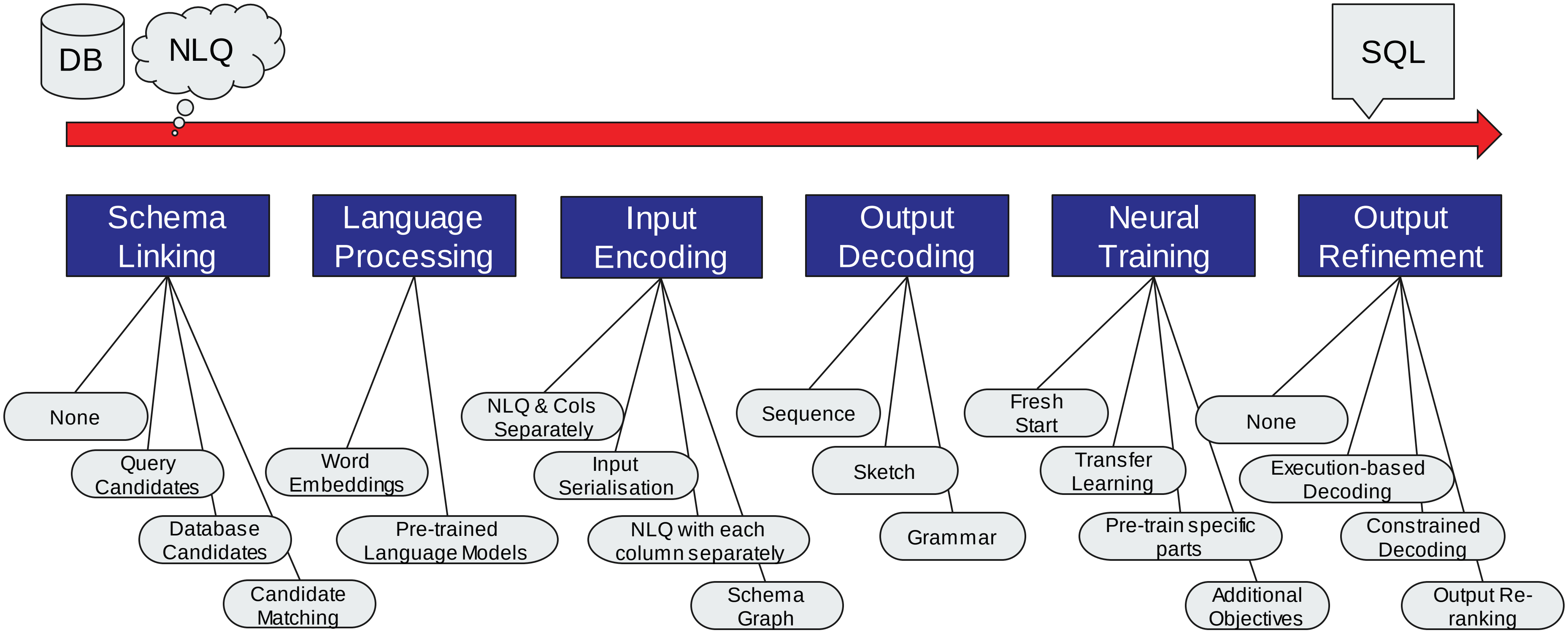
- Models with sequence-based decoders are becoming all the more **powerful** (e.g., T5)
- However, their main drawback is their proneness to **syntactic and grammatical errors**
- Constrained decoding works to **prevent** sequence-based models from producing **erroneous queries**

- PICARD proposes a novel method for incrementally parsing and constraining auto-regressive decoders
  - For each token prediction, PICARD examines the top-*k* most probable tokens
  - If any of the *k* tokens would result in a **grammatical error**, it is discarded
  - If any of the *k* tokens contain an **attribute that is not present in the DB**, it is discarded

Output Refinement: Discriminative Re-ranking

- The nature of neural networks allows us to **extract multiple predictions** for the same NLQ
- Maybe the highest-ranked by the network is not always the correct
- Global-GNN proposes an additional network to **re-rank the *k* highest-ranked** predictions



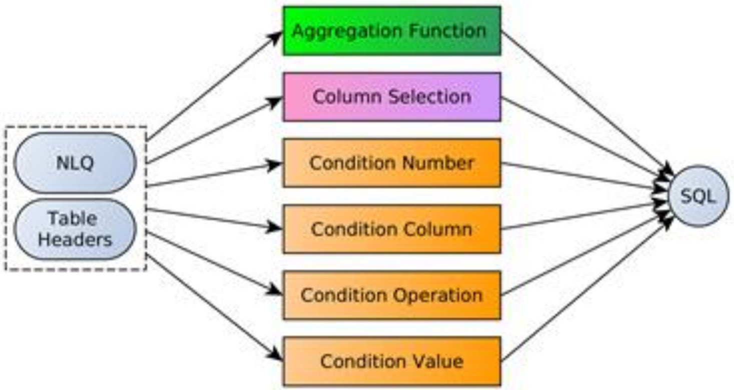


Taxonomy Overview of a Deep Learning Text-to-SQL system

Key Text-to-SQL Systems

SQLNet

- Sketch-based decoding
  - All predicted queries follow the query sketch
  - Separate networks predict different parts of the SQL query
- Separate encoding for NLQ and Table Headers
  - LSTM encoders with GloVe Embeddings
  - Shared across for all sub-networks



SELECT <AGG> <COLUMN>

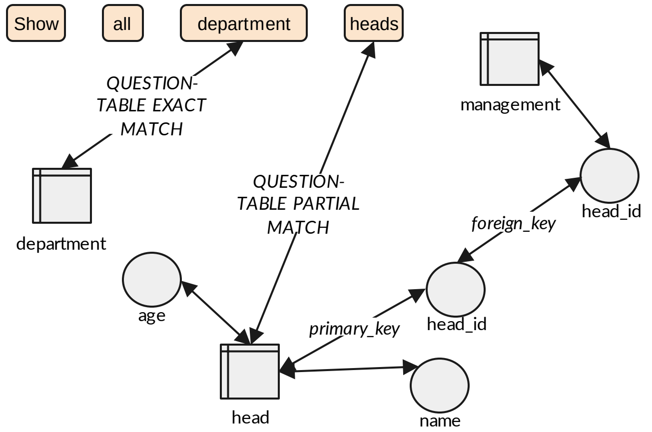
(WHERE <COLUMN> <OP> <VALUE>

(AND <COLUMN> <OP> <VALUE>) *) ?

Schema Linking	NL Representation	Input Encoding	Output Decoding	Neural Training	Output Refinement
None	Word Embeddings	Separately	Sketch-based	Fresh Start	None

RAT-SQL - Encoder

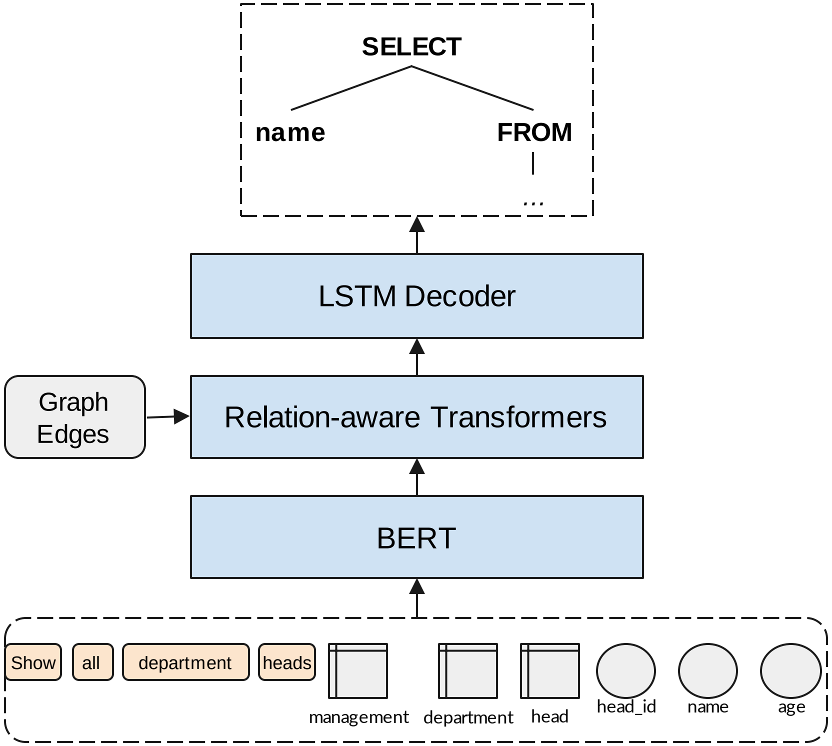
- **Question-contextualized schema graph**
- **Schema** nodes and **NLQ** word nodes
- Edges are **relations** between them from:
  - **Schema relations**
  - **Name-based Linking** (exact or partial n-gram match)
  - **Value-based Linking** (through DB indices or textual search)
- Encoding with GloVe & LSTM or BERT



Schema Linking	NL Representation	Input Encoding	Output Decoding	Neural Training	Output Refinement
n-gram match, indices	Encoder-only PLM	Graph encoding	Grammar-based	Transfer Learning	None

RAT-SQL - Decoder

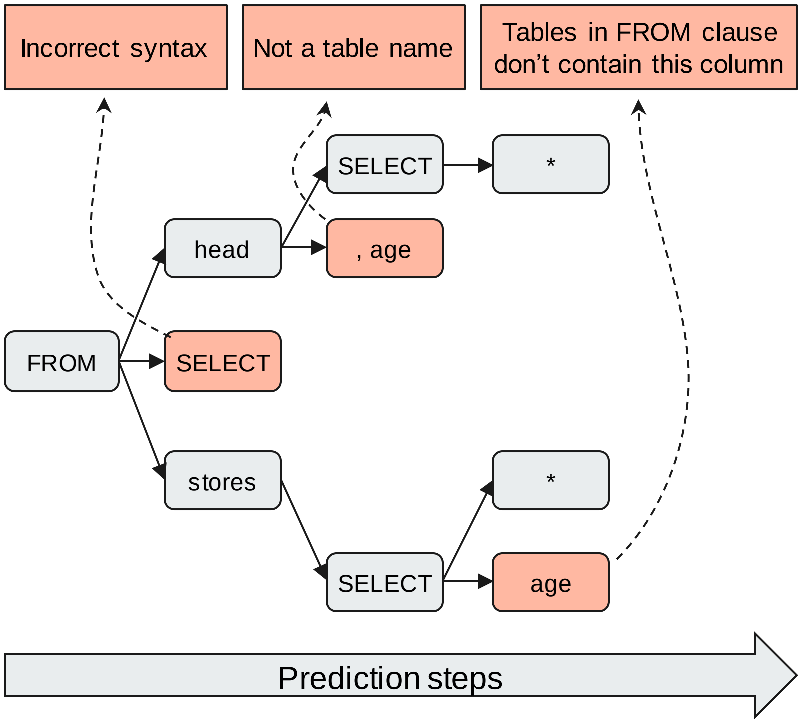
- Specially modified Transformers, for **relation-aware self-attention**, biases the network towards known relations (edges)
- SQL generation as an AST, by predicting a sequence of **decoder actions**
    - Uses a similar **LSTM decoder** to IRNet



Schema Linking	NL Representation	Input Encoding	Output Decoding	Neural Training	Output Refinement
n-gram match, indices	Encoder-only PLM	Graph encoding	Grammar-based	Transfer Learning	None

PICARD

- PICARD is a **constraining technique** for autoregressive decoders of language models
  - Checks for spelling, syntax and grammar errors
  - Checks for availability of used attributes
  - Checks the use of correct aliases
- Tackles the **drawbacks of sequence-based decoders**
- Manages to reach the **top of the Spider** leaderboard in combination with **T5-3B**



Schema Linking	NL Representation	Input Encoding	Output Decoding	Neural Training	Output Refinement
None	Enc-Dec PLM	Serialisation	Sequence-based	Transfer Learning	Constrained Decoding

Challenges and Research Opportunities in Text-to-SQL

Better Text-to-SQL Benchmarks

Researchers tend to rely only on the Spider benchmark for evaluating their systems, ignoring its drawbacks:

- ✕ Databases and queries created specifically for evaluating text-to-SQL systems
  - They do not have the complexity of actual databases used in academia and industry
  - The created DBs contain very little amounts of data
- ✕ Small number of examples for training and evaluation
- ✕ No fine-grained categorisation of different query types, besides “easy”, “medium”, “hard”

- Newer systems can currently reach up to 80% accuracy on Spider
- It’s high time we set new standards:
  - ✓ Create benchmarks using real-world use cases and DBs
  - ✓ Ask real users to provide the queries that they would want to ask the DB
  - ✓ Include in-depth categories to better understand each system’s capabilities

e.g., robustness on synonyms, misspellings, missing info, etc.

Technical Feasibility of Text-to-SQL Systems

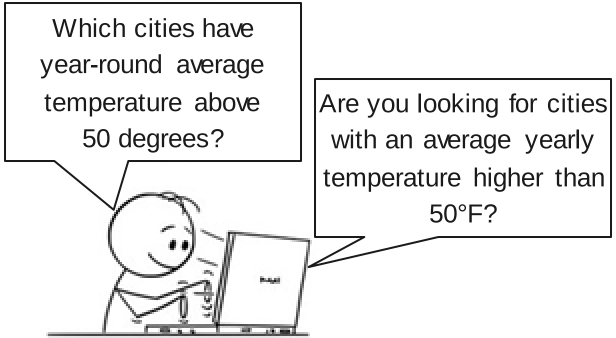
- A lot of breakthroughs have been made by using more and more intricate methods
- However, these techniques are often unrealistic for real-life applications
  - ✕ Large PLMs → Expensive infrastructure and slow predictions
  - ✕ Extensive schema linking → Very slow for large DBs
  - ✕ Constrained decoding → Expensive infrastructure and slow predictions

- A lot of room for contributions in making existing techniques more robust
  - Better performance without very large PLMs
  - Optimised schema linking techniques
- It is necessary to evaluate models not only by their accuracy, but also:
  - Their size
  - Their computing requirements
  - Their prediction latencies

The SQL-to-Text Problem

The SQL-to-Text Problem

- Essential for explaining queries to non-technical users in a NLIDB
  - To verify the prediction of a Text-to-SQL system
  - To allow the user to choose between multiple predictions of a Text-to-SQL system
- Also useful for:
  - Automatic comment generation
  - Helping technical users understand complex queries faster
  - Data augmentation for Text-to-SQL



Challenges: From the NL side

- Generated NL explanations must:
  - Be fluent, coherent and human-like

    “Show year average city temperatures Europe” → “Show the average yearly temperatures of European cities”

  - Avoid unnecessary repetition

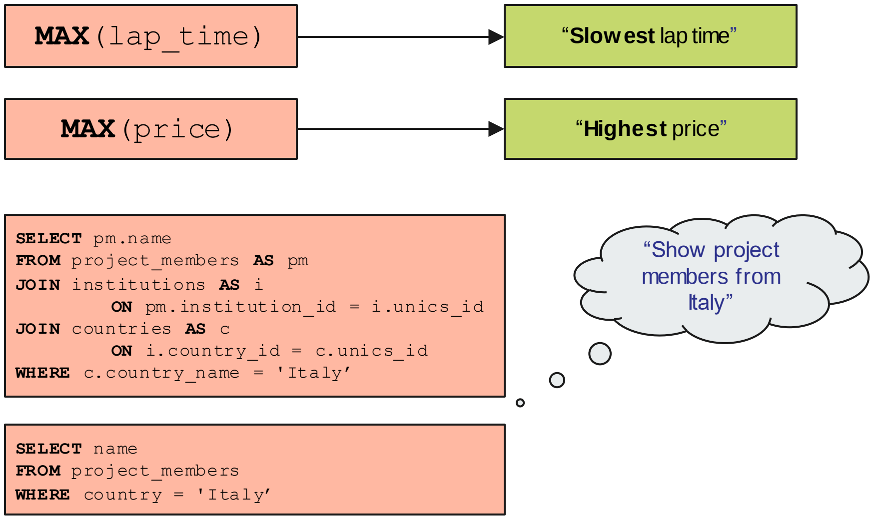
    “Show projects that start after 2014 and before 2020” → “Show projects starting between 2014 and 2020”

  - Avoid over-complications

    “Show me actors that play in a movie and other actors w ho also play in the same movie” → “Show actors that have played in the same movie”

Challenges: From the SQL side

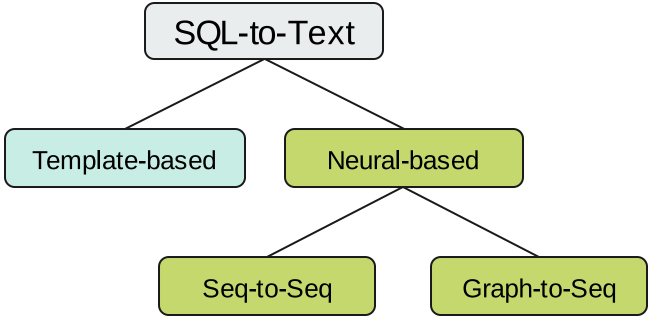
- Using the correct vocabulary based on the DB domain
- Capturing the semantics of complex SQL queries
  - Some parts of the query might not need to be explicitly verbalised
  - The same semantics might be expressed differently, in DBs with different schemas



SQL-to-Text Approaches and Key Systems

SQL-to-Text Approaches

- Has seen relatively less attention compared to the fast-paced Text-to-SQL field
  - Only a handful of deep learning systems
  - No established benchmark or metric
- Earlier approaches used templates and rules to construct query explanations
- Recently, a few deep learning approaches have sprung, mostly motivated by data augmentation for Text-to-SQL



SQL-to-Text: Template-based Approaches

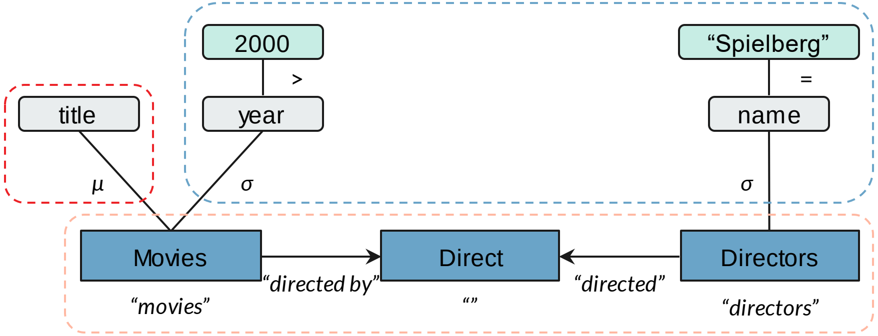
- A query graph is created based on the input query
- A set of templates for each part of DB is provided
- The query explanation is created by traversing the query graph and using the appropriate templates

- ✔ Very precise, since they verbalise all parts of the query
- ✘ A new set of templates is needed when moving to a new DB
- ✘ The query explanations are not fluent and realistic

An example of template-based SQL-to-Text

```
SELECT m.title
FROM Movies m
JOIN Direct r ON m.id=r.movie_id
JOIN Director d ON r.director_id=d.id
WHERE d.name='Spielberg' AND m.year=2000
```

"Find the titles of movies that have been directed by directors. Return results only for movies whose release year is 2000 and directors whose name is Spielberg."



SQL-to-Text: Neural-based Approaches

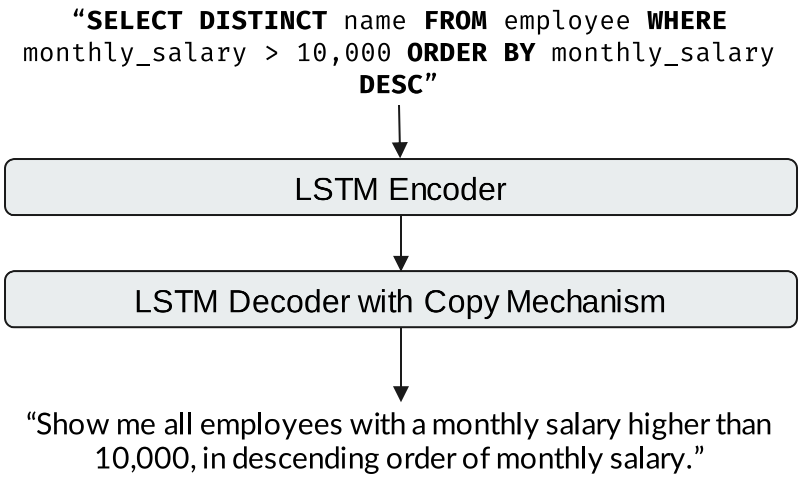
- ☑ Can produce much more fluent and natural explanations
- ☑ Are easier to generalise to unseen DBs, even without human labour
- ✕ Can not guarantee the precision of their explanations

- Two main categories of deep learning SQL-to-Text:
  - Sequence-to-Sequence
  - Graph-to-Sequence
- A relatively unexplored field

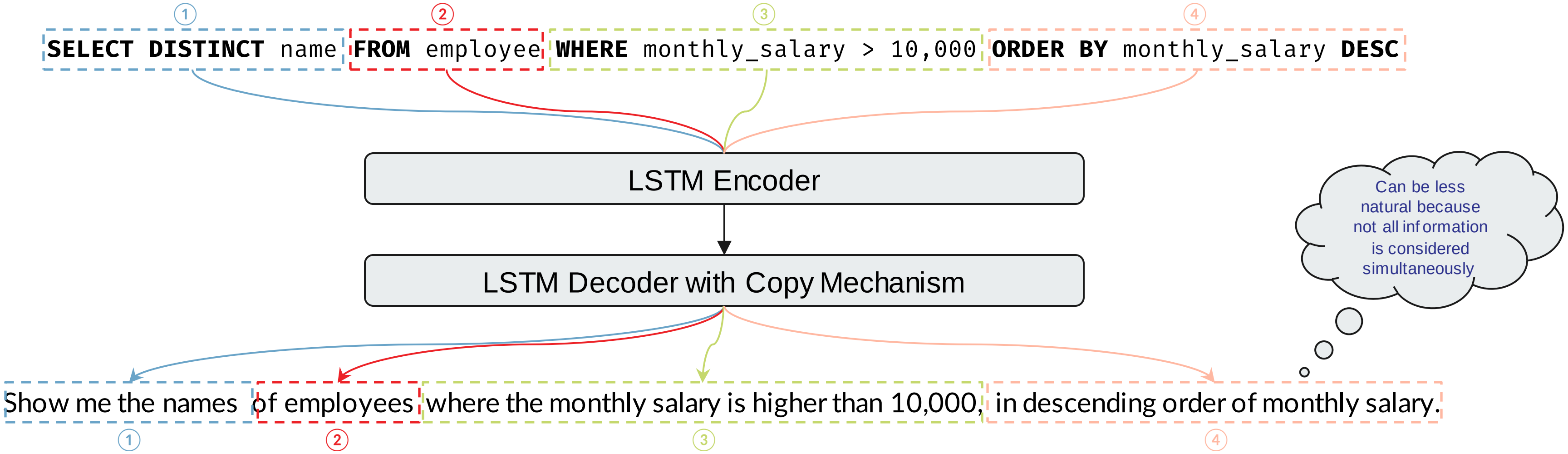
Model	**WikiSQL** (BLEU)	**Spider** (BLEU)
Seq-to-Seq [32]	18.40	-
Graph-to-Seq [33] (GNN)	28.70	-
Graph-to-Seq [34] (RGT)	31.20	28.84

SQL-to-Text: Sequence-to-Sequence

- The SQL query is decoded as a text sequence
- The explanation is generated using an RNN or Transformer decoder
- Similarly to any other translation task
- Does not take advantage of the inherent structure of SQL

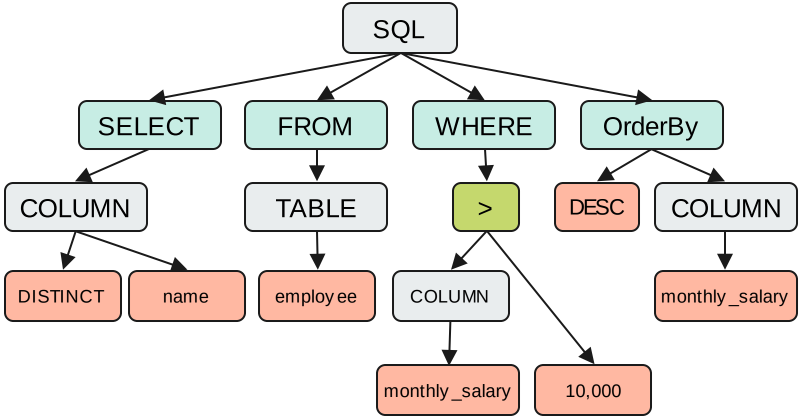


Hierarchical Sequence-to-Sequence



Non-hierarchical: *Show me all employees with a monthly salary higher than 10,000, in descending order of monthly salary.*

SQL-to-Text: Graph-to-Sequence



"Show me all employees with a monthly salary higher than 10,000, in descending order of monthly salary."

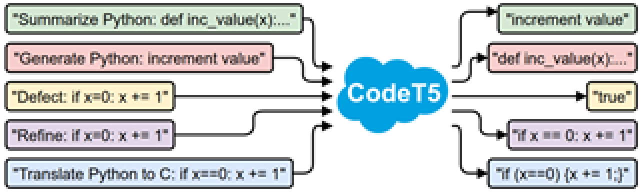
- The SQL query is encoded as a graph or as a tree
  - Using GNNs, or Graph Transformers
- The explanation is generated using a RNN, or Transformer-based decoder
- Notice the differences between this representation and the one used by template-based approaches

Challenges and Research Opportunities in SQL-to-Text

SQL-to-Text: The use of PLMs

- The success of PLMs is quickly extending to code-related tasks:
    - Code Summarisation
    - Code Generation
    - Code Translation
    - Code Refinement
    - Defect/Vulnerability Detection
    - Clone Detection
    - Semantic Similarity
- Models such as CodeBERT, CodeT5 , and PLBART

- In this case researchers investigate:
    - Which pre-training tasks help the most
    - How to formulate their inputs
    - Model size and hyper-parameter tuning
- Not a lot of research on their value for the SQL-to-Text task yet



A Metric for Query Explanations

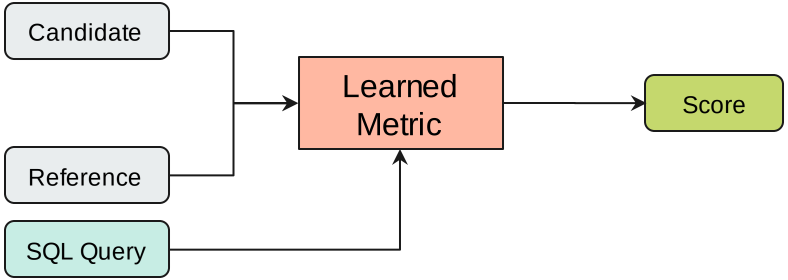
There are no tailor-made metrics for query explanations

Automatic translation metrics are not robust to vocabulary differences and do not take the query into account

Ground Truth	Prediction		BLEU	chrF	METEOR
How many singers do we have?	How many songs do we have?	✗	48.89	68.49	80.66
	What is the number of singers?	✓	7.80	24.15	0.00
Tell me the age of the oldest dog.	Tell me the age of the youngest dog.	✗	66.06	72.83	86.47
	How old is the eldest dog?	✓	5.66	37.42	16.12

Using Learned Metrics for Query Explanations

- It is evident that a robust metric for query explanations should:
  - Take semantic similarity into account, not just common words and n-grams
  - Work well for short text inputs



- Inspiration from learned metrics:
  - Use cosine similarity on sentence embeddings produced by a PLM (e.g., BERTScore)
  - Train a PLM to predict a score on its own (e.g., BLEURT)
- Design a new model using a PLM
  - Take advantage of the SQL query as well

What is a query explanation?

- Recent works focus on applying deep learning for SQL-to-Text
- However, no discussion has been made on how a query explanation should be
- A previous study [44] identifies three expression types:
  - Statement
  - Question
  - Command

"Employees with a monthly salary higher than 10,000."

"Which employees earn a monthly salary higher than 10,000?"

"Show me all employees with a monthly salary higher than 10,000."

- There are additional aspects of query explanations that need to be better defined
- At the end of the day, a better understanding of the problem helps us design better systems and metrics

What is a query explanation: Level of Detail

- How much detail is needed to explain a query?
  - Too much detail can be tiring for the user
  - Sometimes a very precise explanation is needed
- Which factors determine the needed level of detail?
  - User preferences?
  - DB Domain (e.g., medical research might need higher detail)?
  - What else?

```
SELECT name, location, district
FROM shop
ORDER BY number_products DESC
```

- “Show me the shops, ordered by their number of products”
- “Show me the name, location and district of all shops, in descending order of number of products”

What is a query explanation: More Examples

```
SELECT T2.name , COUNT(*)
FROM concert AS T1
JOIN stadium AS T2
ON T1.stadium_id = T2.stadium_id
GROUP BY T1.stadium_id
```

- “How many concerts took place in each stadium?”
- ”Show me the number of concerts per stadium along with the name of the stadium”

```
SELECT *
FROM project
WHERE start_year > 2014
```

- “Show me projects that started after 2014”
- “Show information about projects starting after 2014”
- “Show everything about projects that start after 2014”

Creating a SQL-to-Text Dataset

- Currently no dataset/benchmark created specifically for SQL-to-Text
- All proposed systems use Text-to-SQL datasets
- This is also highly connected to the lack of an established metric for the problem

- Proposing a common benchmark for SQL-to-Text could bring a burst of research to the field similarly to the Spider dataset
  - Establish a common benchmark for a fair comparison
  - Provide multiple explanations for each SQL input
  - Variations in style (question, command, etc.) and detail
  - Interesting categories to better evaluate system performance

Time for a coffee break!

We will resume at 15:30

Part 1

Text-to-SQL

1. The Text-to-SQL problem
2. Benchmarks
3. A Taxonomy for Deep Learning Text-to-SQL Systems
4. Key Systems
5. Research Challenges

Part 2

SQL-to-Text

1. The SQL-to-Text problem
2. Challenges
3. Key Systems
4. Research Challenges

30 min.

Part 3

Data-to-Text

1. What is Data-to-Text
2. Subfields of Data-to-Text
3. Table-to-Text
4. Graph-to-Text
5. Evaluation
6. Research Challenges

Part 4

Bringing it all together

1. What do we mean?
2. Why is it not trivial?
3. Challenges
4. Demo

The Data-to-Text Problem

What is Data-to-Text?

Definition: Translating information from a structured form to natural language

Carl Friedrich Gauss	
Born	Johann Carl Friedrich Gauss 30 April 1777 Brunswick, Principality of Brunswick-Wolfenbüttel, Holy Roman Empire
Died	23 February 1855 (aged 77)

Carl Friedrich Gauss was born on the 30th of April 1777 in Brunswick and died on 23th of February 1855 at the age of 77.

The generated text must feel fluent but remain faithful to the truth

Some parts should be omitted

The model must understand the input structure

Why Data-to-Text?

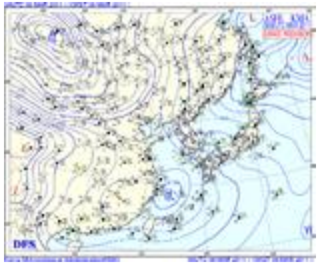
- ➔ Automating and assisting tedious report creation
- ➔ Explanation of data that need expertise to understand
- ➔ Create insights of large amounts of data, not interpretable by a human

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AS …
Heat	11	12	103	49	47	27
Hawks	7	15	95	43	33	20

PLAYER	AS	RB	PT	FG	FGA	CITY …
Tyler Johnson	5	2	27	8	16	Miami
Dwight Howard	4	17	23	9	11	Atlanta
Paul Millsap	2	9	21	8	12	Atlanta
Goran Dragic	4	2	21	8	17	Miami
Wayne Ellington	2	3	19	7	15	Miami
Dennis Schroder	7	4	17	8	15	Atlanta
Rodney McGruder	5	5	11	3	8	Miami
Thabo Sefolosha	5	5	10	5	11	Atlanta
Kyle Korver	5	3	9	3	9	Atlanta
…						

Box-score statistics of a basketball game

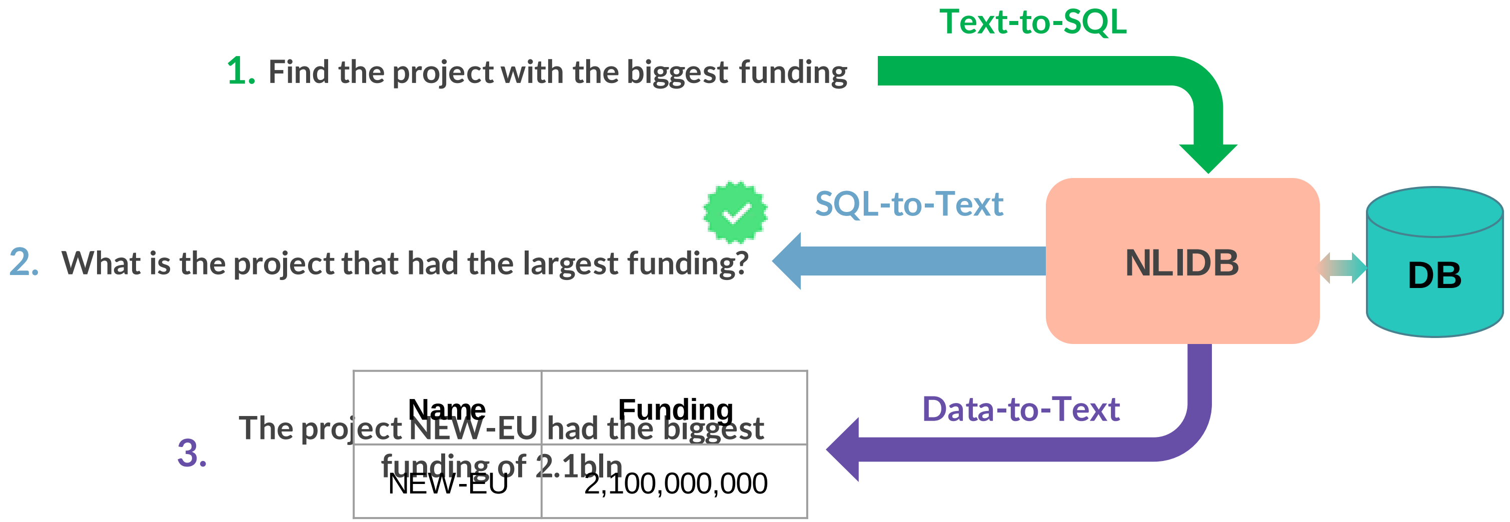
The Atlanta Hawks defeated the Miami Heat , 103 - 95 , at Philips Arena on Wednesday . Atlanta was in desperate need of a win and they were able to take care of a shorthanded Miami team here . Defense was key for …



Tomorrow expect strong SW winds on the coasts of South Korea

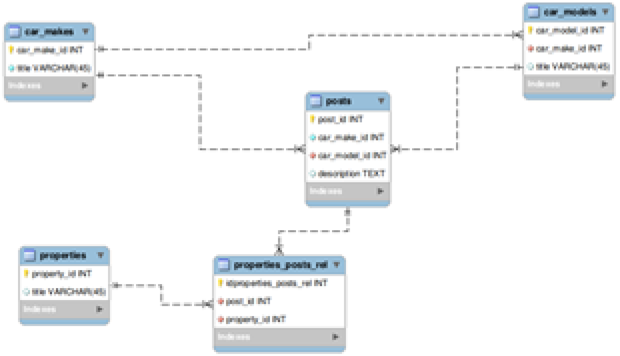
Why Data-to-Text in a NL Interface?

→ The whole interaction of the user would only use natural language.



Why Data-to-Text in an NLIDB?

→ **Explain the schema of the database with natural language**



Cars Simple DB

"This database provides info about car makers …"

- ➔ Automatic annotation of a catalog of databases.
- ➔ Faster database schema familiarization

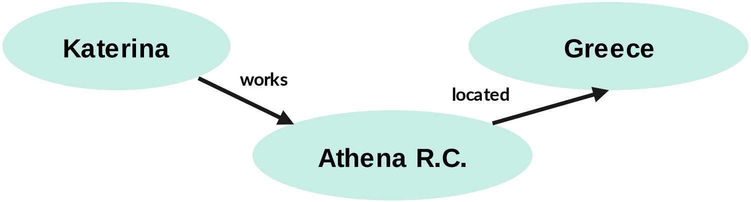
Data-to-Text Sub-fields

Table-to-Text

Name	Age	Height
Katerina	25	1.90

“Katerina is 25 years old and has a height of 1.90”

Graph-to-Text



“Katerina is employed by Athena R.C. which is located in Greece.”

The Table-to-Text Problem

Table-to-Text

Given a table, generate text that expresses the information of the whole table or parts of it.

Movie	Actor	Director
A Beautiful Mind	Russell Crowe	Ron Howard

Table-to-Text

"The movie A Beautiful Mind starring Russell Crowe was directed by Ron Howard."

Datasets

Year	Dataset	Domain	Examples
2009	WEATHERGOV	Weather	29,528
2016	WIKIBIO	Wikipedia Bios	728,357
2017	E2E	Restaurants	51,426
2017	ROTOWIRE	Basketball	4,826
2018	ESPN	Basketball	15,054
2018	Wikiperson	Wikipedia Bios	310,655
2019	ROTOWIRE-MODIFIED	Basketball	3,734
2019	MLB	Basketball	26,304
2019	Rotowire-FG	Basketball	7,476
2020	LOGICNLG	Wikipedia	37,015
2020	ToTTo	Wikipedia	136,161
2021	WIKITABLET	Wikipedia	1.5M
2021	SciGen	Scientific	1,300
2021	TWT	Wikipedia	128,268 and 49,417
2022	Hitab	Wikipedia	10,686

✗ **Way too domain specific**
- Many datasets are domain specific leading to models overfitting.

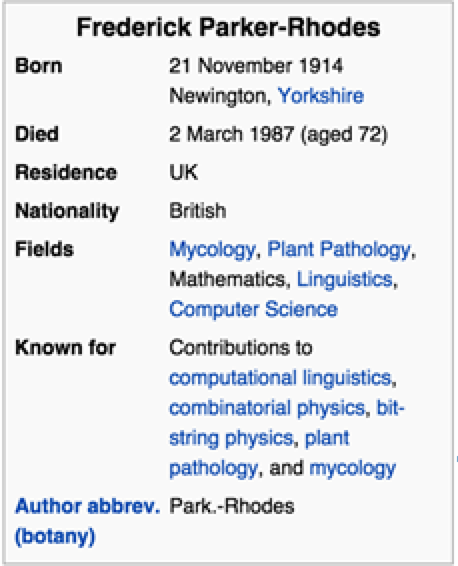
✗ **Not a unified structure format**
- A model architecture on a dataset is not transferable to other datasets.

Most influential datasets, which their challenges caused many important innovations in Table-to-Text.

WIKIBIO - 2016

➔ **Size:** 728K infoboxes

It comprises of all the the biography articles in the WikiProject along with their infoboxes.



Frederick Parker-Rhodes	
Born	21 November 1914 Newington, Yorkshire
Died	2 March 1987 (aged 72)
Residence	UK
Nationality	British
Fields	Mycology, Plant Pathology, Mathematics, Linguistics, Computer Science
Known for	Contributions to computational linguistics, combinatorial physics, bit-string physics, plant pathology, and mycology
Author abbrev. (botany)	Park.-Rhodes

"Frederick Parker-Rhodes (21 March 1914 – 21 November 1987) was an English linguist, plant pathologist, computer scientist, mathematician, mystic, and mycologist."

Average generated text length: 53.1 words

Average infobox length: 26.1 words

ROTOWIRE - 2017

➔ **Size:** 4.9K statistics-report pairs in total

A dataset of NBA basketball game statistics paired with their human-written reports.

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AST	...
Pacers	4	6	99	42	40	17	...
Celtics	5	4	105	44	47	22	...

PLAYER	H/V	AST	RB	PTS	FG	CITY	...
Jeff Teague	H	4	3	20	4	Indiana	...
Miles Turner	H	1	8	17	6	Indiana	...
Isaiah Thomas	V	5	0	23	4	Boston	...

The **Boston Celtics** defeated the host **Indiana Pacers 105-99** at Bankers Life Fieldhouse on Saturday. In a battle between two injury-riddled teams, the Celtics were able to prevail...

Average number of records: 628

Average generated text length: 805 words

A ROTOWIRE version with no leaks between train and test was created called: *SportSett:Basketball*

ToTTo - 2020

➔ **Size:** 135K highlighted tables

Given a Wikipedia table and a set of highlighted table cells, produce a one-sentence description

Table Title: Gabriele Becker
Section Title: International Competitions
Table Description: None

Year	Competition	Venue	Position	Event	Notes
Representing Germany					
1992	World Junior Championships	Seoul, South Korea	10th (semis)	100 m	11.83
1993	European Junior Championships	San Sebastián, Spain	7th	100 m	11.74
			3rd	4x100 m relay	44.60
1994	World Junior Championships	Lisbon, Portugal	12th (semis)	100 m	11.66 (wind: +1.3 m/s)
			2nd	4x100 m relay	44.78
1995	World Championships	Gothenburg, Sweden	7th (q-finals)	100 m	11.54
			3rd	4x100 m relay	43.01

"Gabrielle Becker competed at the 1995 World Championships both individually and on the relay."

ToTTo manages to be:

- ✓ Big (135K)
- ✓ Diverse
  - Sports
  - Countries
  - Politics
  - …
- ✓ High quality

But how was it created?

ToTTo - Creation

Page Title: Quentin Tarantino

Movie	Year	Budget
Django	2012	426mil
Kill Bill	2003	30mil

Step 1: Crawl a table from Wikipedia

Step 2: Retrieve a text passage that overlaps

"After 2010 he directed Django in 2012."

Step 3: Highlight corresponding cells

Step 4: Remove parts that cannot be inferred the table

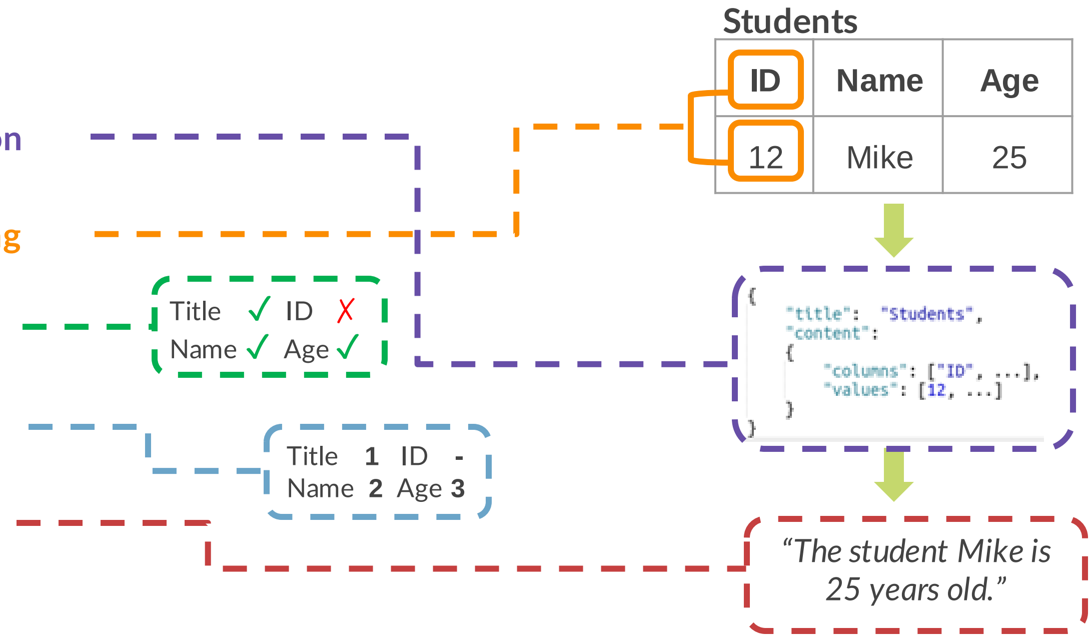
"~~After 2010~~ he directed Django in 2012."

Step 5: Make the sentence independent

"Quentin Tarantino directed Django in 2012."

Challenges of Table-to-Text Systems

- ➔ Table representation
- ➔ Table understanding
- ➔ Content selection
- ➔ Content planning
- ➔ Text generation



Seen Challenges of Table-to-Text Systems

Language processing

- LSTM Neural Networks (1995) 🔗 [5]
- Word Embeddings
    - One-hot Embeddings
    - Word2Vec (2013) 🔗 [6]
    - GloVe (2014) 🔗 [7]
- The Transformer (2017) 🔗 [9]
- The rise of language models
    - BERT (2018) 🔗 [10]
    - RoBERTa (2019) 🔗 [11]
    - TaBERT (2020) 🔗 [12]
    - GraPPa (2020) 🔗 [13]
    - BART (2020) 🔗 [28]
    - T5 (2020) 🔗 [29]

✓✓ Seen 1:50 PM

Seen Challenges of Table-to-Text Systems

Neural Training

1. **Fresh Start:** Train the network from scratch
   - The most common approach for neural networks
2. **Transfer Learning:** First pre-train on a generic task, then fine-tune for text-to-SQL
   - The Computer Vision and NLP domains have proven its power
   - Has seen widespread use with the introduction of Transformer-based PLMs

✓✓ Seen 1:58 PM

Challenges of Table-to-Text Systems

→ Table representation

How will we represent the table and its metadata in a machine readable format?

- WikiBIO
- ROTOWIRE
- ToTTo

Students

ID	Name	Age
12	Mike	25


```
{
    "title":  "Students",
    "content":
    {
        "columns": ["ID", ...],
        "values": [12, ...]
    }
}
```


"The student Mike is 25 years old."

[57] Table-to-text Generation by Structure-aware Seq2seq Learning (2017)
[58] TableGPT: Few-shot Table-to-Text Generation with Table Structure Reconstruction and Content Matching (2020)

ATHENA

WIKIBIO Representation

Fields	Values
name	George Mikell
birthname	Jurgis Mikelaitis
birthdate	4 April 1929 (age 88)
birthplace	Bildeniai, Lithuania
nationality	Lithuanian, Australian
occupation	Actor, writer
years active	1957–present
known for	The Guns of Navarone The Great Escape

Wikipedia infobox of George Mikell

1. **Separate each value word**
   "George Mikell" -> ["George", "Mikell"]
1. **Add field name ("name")**
2. **Add position values (1, 2)**

eg. "George": (name, 1, 2)

What field is described by "George"

- **Ordering**
- **Proximity to the end**

word	*Field embedding*
George	(name, 1, 2)

Newer language models (eg. GPT) can handle whole phrases allowing for simplifications.

name	George Mikell
birthname	Jurgis Mikelaitis

["name **is** George Mikell", "birthname **is** Jurgis Mikelaitis"]

ROTOWIRE Representation

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AST	...
Pacers	4	6	99	42	40	17	...
Celtics	5	4	105	44	47	22	...

PLAYER	H/V	AST	RB	PTS	FG	CITY	...
Jeff Teague	H	4	3	20	4	Indiana	...
Miles Turner	H	1	8	17	6	Indiana	...
Isaiah Thomas	V	5	0	$\boxed{23}$	4	Boston	...
Kelly Olynyk	V	4	6	16	6	Boston	...
Amir Johnson	V	3	9	14	4	Boston	...
...	...	...	...	...	...	...	

Data records of the results of NBA basketball game

Each record consists of:

1. type (PTS)
2. entity (Isaiah Thomas)
3. value (23)
4. home or visitor (V)

PTS, Isaiah Thomas, 23, V

Value	Entity	Type	H/V
Boston	Celtics	TEAM-CITY	V
Celtics	Celtics	TEAM-NAME	V
105	Celtics	TEAM-PTS	V
Indiana	Pacers	TEAM-CITY	H
Pacers	Pacers	TEAM-NAME	H
99	Pacers	TEAM-PTS	H
42	Pacers	TEAM-FG_PCT	H
22	Pacers	TEAM-FG3_PCT	H
5	Celtics	TEAM-WIN	V
4	Celtics	TEAM-LOSS	V
Isaiah	Isaiah_Thomas	FIRST_NAME	V
Thomas	Isaiah_Thomas	SECOND_NAME	V
23	Isaiah_Thomas	PTS	V

Final record list

✓ We do not lose any information of the original table

✗ Leads to huge length of inputs reaching the token limit of PLMs

ToTTo Representation

Table Title: Cristhian Stuani
Section Title: International goals

No.	Date	Venue	Opponent	Result
2	13 November 2013	Amman International Stadium, Amman, Jordan	Jordan	5-0

ToTTo table extracted from Wikipedia

XML-like ToTTo Representation

<page_title> Christian Stuani </page_title>
<section_title> International goals </section_title>
<table>
<cell> 2. <col_header> No. </col_header> </cell>
<cell> 13 November 2013 <col_header> Date </col_header></cell>

...

</table>

✓ Explicit description of what each value represents
✗ Too verbose with unused info

Column-Value ToTTo Representation

"No. 2 Date 13 November 2013 Venue Amman... Opponent Jordan Result 5-0"

✓ Concise
✓ No need to understand special syntax (eg. XML)
✗ A lot of implicit info

Challenges of Table-to-Text Systems

→ Table representation

→ Table understanding

How will the model understand the correlations that exist in a table?

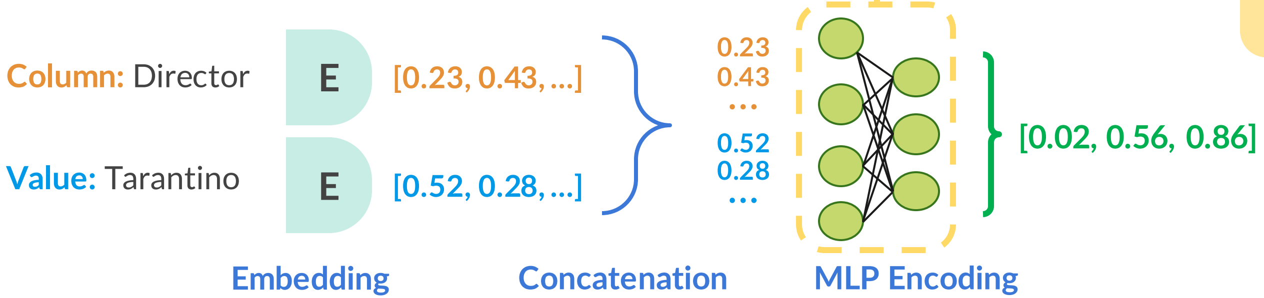
Students

ID	Name	Age
12	Mike	25


```
{
    "title":  "Students",
    "content":
    {
        "columns": ["ID", ...],
        "values": [12, ...]
    }
}
```

Table understanding: MLP Encoder

The MLP is trained end-to-end with the rest of the Table-to-Text architecture



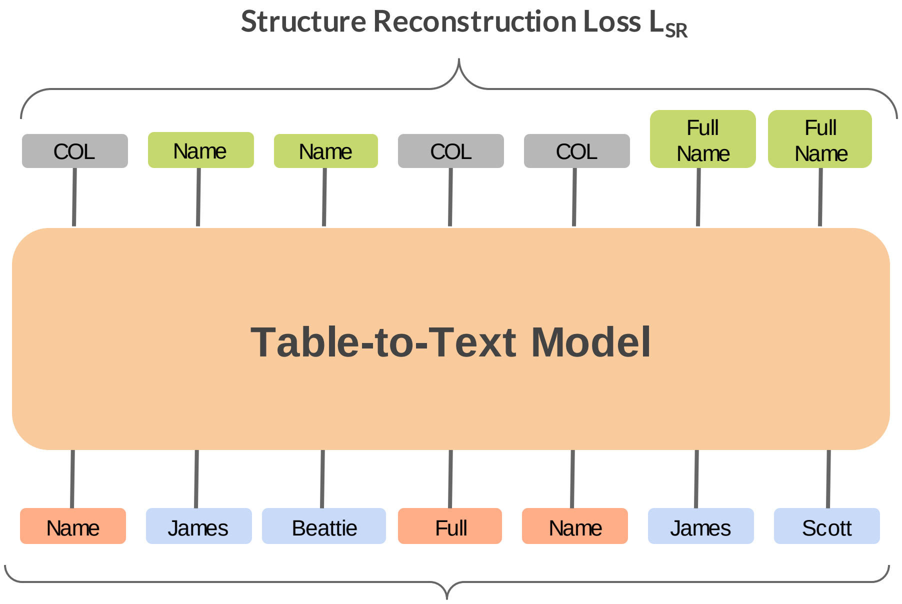
✓ **Simplistic**
Simple architecture to understand and implement.

✓ **Flexible**
If the structure of the input changes we can easily adapt the architecture.

✗ **Gradient vanishing prone**
There are no guarantees that meaningful gradients will reach the MLP due to the end-to-end training.

✗ **Column-value only**
We encode each attribute independently.

Table understanding: Table Reconstruction



Name	Full Name
James Beattie	James Scott

✓ **Column-value encoding**
Forces the model to understand the correlation between column name tokens and values.

✓ **Position encoding**
The model differentiates between James -> Name and James -> Full name

✓ **PLM compatible**
The task is compatible with any model that has an embedding for each of its input tokens, like PLMs.

Table understanding: Numerical Value Representation

Understanding numbers, especially in automatic report generation, is essential for a meaningful verbalisation.

Challenges

The meaning of a number depends on context

Only one player scored **30 points** → **Good performance**

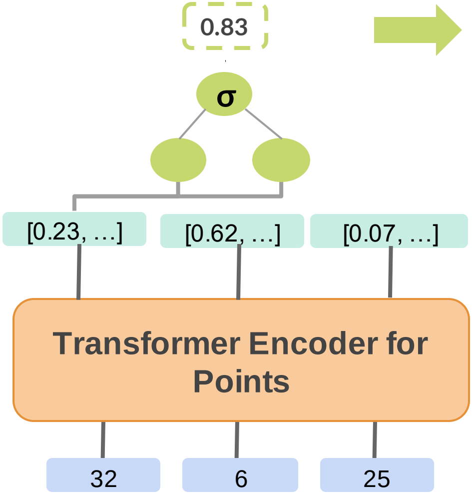
All players scored **30 points** → **Average performance**

The meaning of a number depends on type

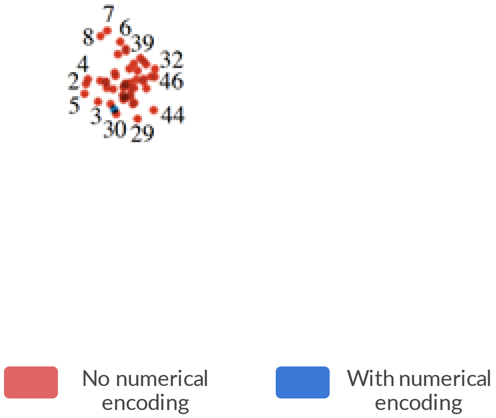
Made **5 assists** → **Good performance**

Scored **5 points** → **Average performance**

Train a transformer encoder to output number contextual number embeddings.



We expect that: r(32) > r(25) >> r(6)



Challenges of Table-to-Text Systems

→ Table representation

→ Table understanding

→ Content selection

Title ✓ ID ✗
Name ✓ Age ✓

Which parts of the table should the model pick for verbalisation?

Students

ID	Name	Age
12	Mike	25


```
{
    "title": "Students",
    "content":
    {
        "columns": ["ID", ...],
        "values": [12, ...]
    }
}
```

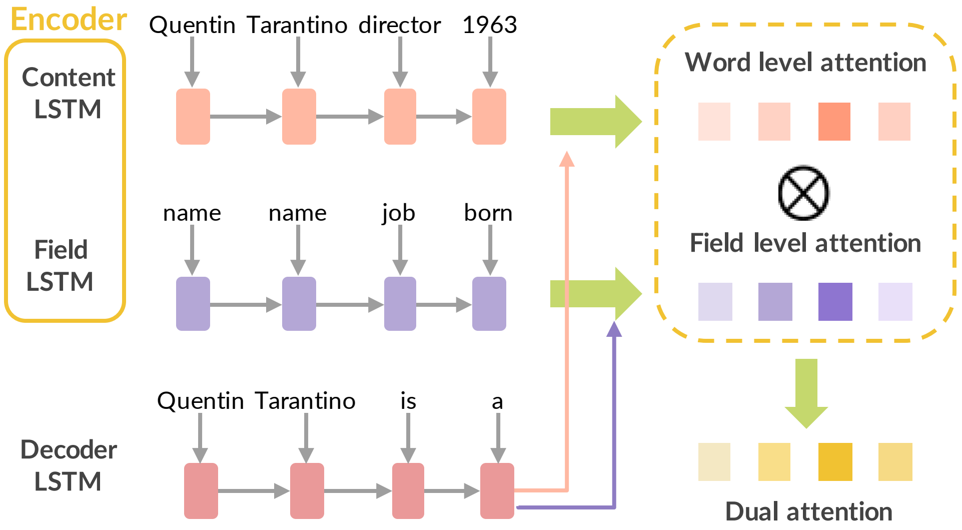

“The student Mike is 25 years old.”

Content selection: Dual attention

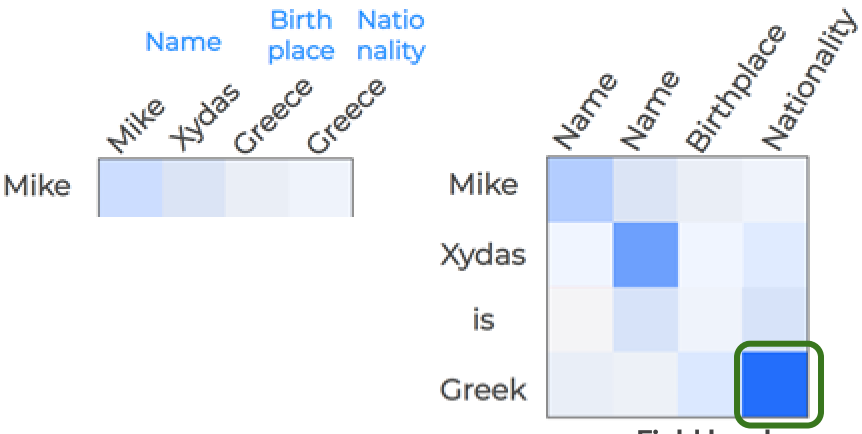
name	job	born
Quentin Tarantino	director	1963

Word level attention: Capturing the semantic relevance between generated tokens and the content information.

Field level attention: Locate the particular field-value record we should focus while generating the next token.



Why do we need both word and field attention?



Word level attention

Field level attention

Content selection: Target prediction

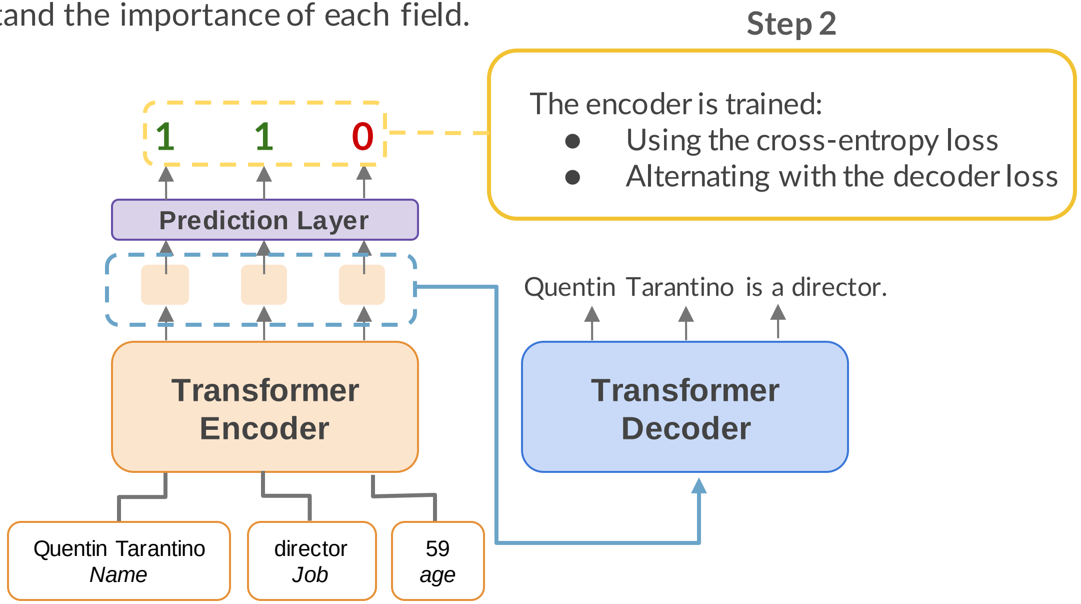
Goal: Explicitly force the model to understand the importance of each field.

Name	Job	Age
Quentin Tarantino	director	59

Target: Quentin Tarantino is a director.

Extract which records appear in the verbalisation

Step 1



Challenges of Table-to-Text Systems

- ➔ Table representation
- ➔ Table understanding
- ➔ Content selection
- ➔ Content planning

Title **1** ID **-**
Name **2** Age **3**

In what order should the picked attributes be verbalised?

Students

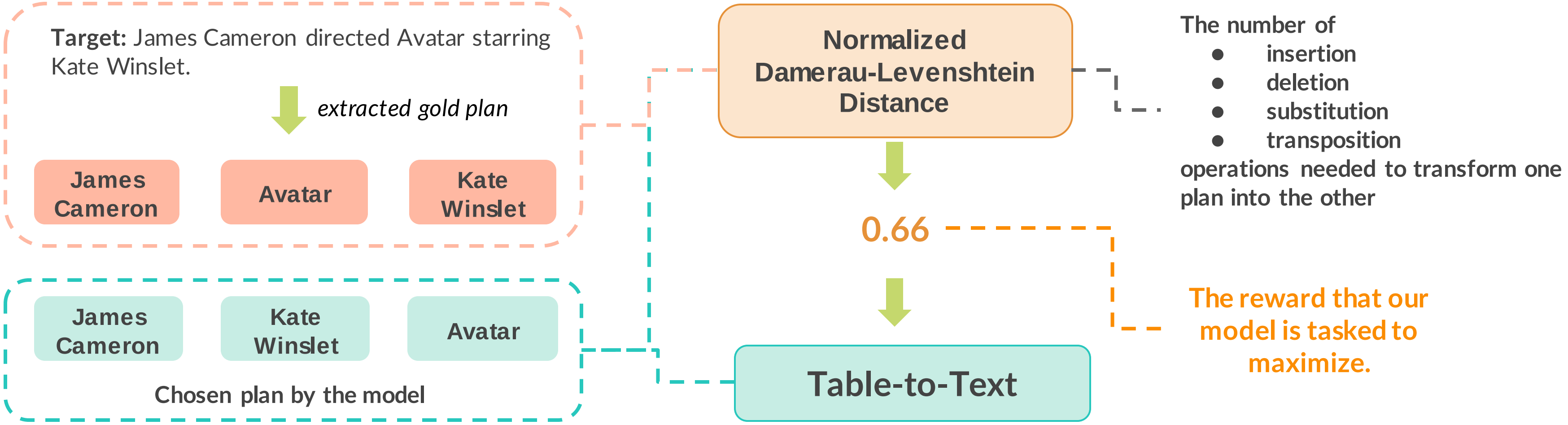
ID	Name	Age
12	Mike	25


```
{
  "title": "Students",
  "content":
  {
    "columns": ["ID", ...],
    "values": [12, ...]
  }
}
```


"The student Mike is 25 years old."

Content planning: Record ordering reward

Optimize the model based on the record order of the produced plan.
Extract the order from ground truth verbalisations and then let the policy gradient and backpropagation do the rest.



Content planning: Linear Chain CRFs

Train a linear chain conditional random field (CRF) to find the optimal ordering between encoded table attributes.

The CRF will optimize the likelihood of plan Y given the content encodings H_C:

$P_{CRF}(Y \mid H_C)$ → P_{CRF} ([3 , 1, 2, ∅] |

1. ***Enc***(Year: 2019)
2. ***Enc***(Title: Dune)
3. ***Enc***(Director: Villeneuve)
4. ***Enc***(Budget: 165 mln))

The true plan (for training) is extracted by the verbalisation

"The movie titled ***Dune***[1] *was directed by* ***Villeneuve***[2] *and released in* ***2019***[3]*."*

∅ indicates the omission of the corresponding token in the content plan, in our example *Budget*.

During inference we find the sequence $Y = \mathrm{argmax}_Y \; P_{CRF}(Y \mid H_C)$

Challenges of Table-to-Text Systems

- → Table representation
- → Table understanding
- → Content selection
- → Content planning
- → Text generation

Given the final plan of the selected attributes how will the model verbalise it on natural language?

Students

ID	Name	Age
12	Mike	25


```
{
    "title":  "Students",
    "content":
    {
        "columns": ["ID", ...],
        "values": [12, ...]
    }
}
```


"The student Mike is 25 years old."

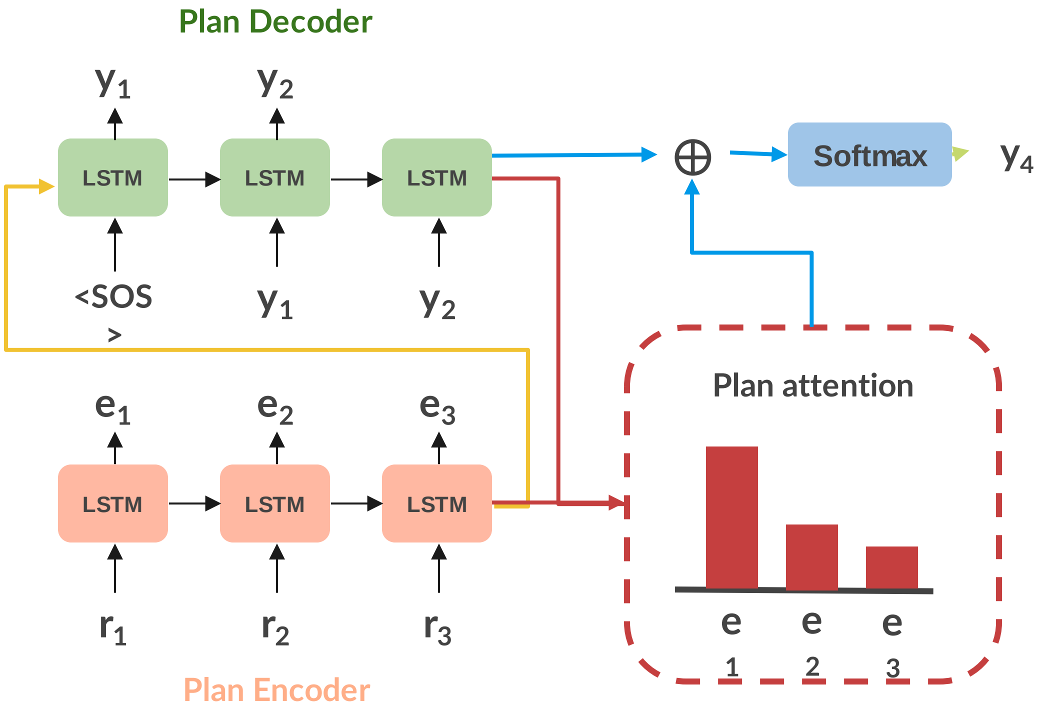
Text generation: LSTM and Attention

Our generation task is not a simple text-to-text task but requires:

- Info about the plan created
- Info about the previous words generated
- Info about the general context of the table

✓ **Interpretable**
The attention can be explored to “debug” our architecture.

✗ **Vocabulary limitation**
If a word we want is outside of our limited vocabulary, it’s impossible to generate it.



Text generation: Copy mechanism

Problem: In the domain of table verbalisation many words have a low frequency.

Many words have a low chance of being included in the vocabulary (~20K words)

Solution: Establish a **copy mechanism** that will directly copy a word from the input without losing the power of generative models.

$$p_t(\tilde{y}) = g_t(\tilde{y}) \odot \alpha_{\langle t, id(\tilde{y})\rangle} + (1 - g_t(\tilde{y})) \odot \beta_t(\tilde{y})$$

The final distribution over all the words (both in the vocabulary and the table records).

A NN that estimates the tradeoff [0, 1] between copying or generating the next token.

Attention over the records of the table.

The softmax distribution over our vocabulary.

Challenges of Table-to-Text Systems

→ Table representation

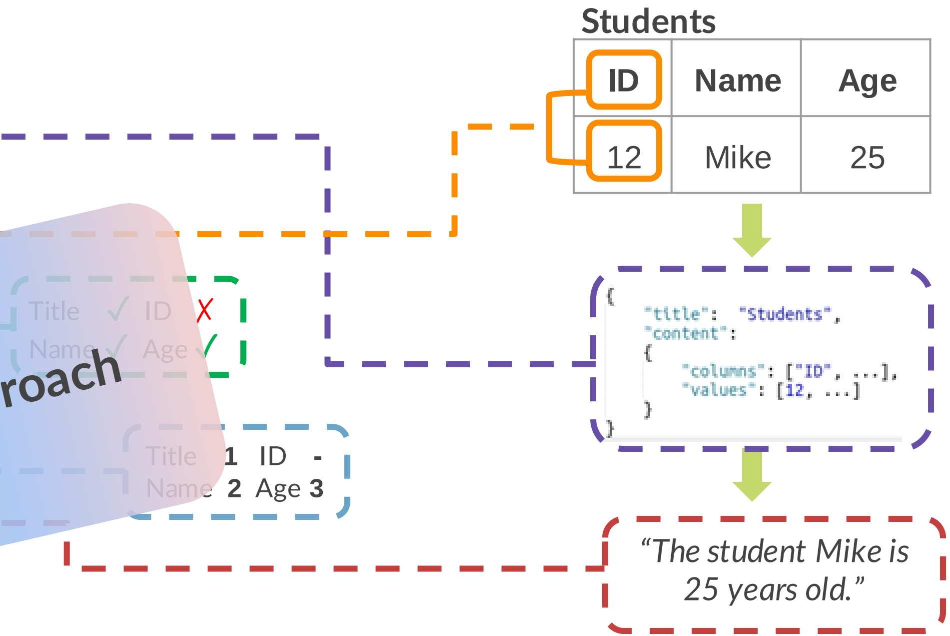
→ Table understanding

→ Content selection

→ Content planning

→ Text generation

End-to-End Approach



End-to-End Solution

Table Title: Cristhian Stuani				
Section Title: International goals				
No.	Date	Venue	Opponent	Result
2	13 November 2013	Amman International Stadium, Amman, Jordan	Jordan	5-0

<page_title> Christian Stuani </page_title>
<section_title> International goals
</section_title>
<table>
<cell>
2. <col_header> No. </col_header>
</cell>
...
</table>

PLM

"On 13 November 2013 Christian Stuani netted the second in a 5-0 win in Jordan."

✓ **Powerful model**
These models have great understanding of text.

✓ **Simple**
Simple to understand and easy to experiment with.

✗ **Hard to modify**
Changing the architecture could lead to the model forgetting text understanding .

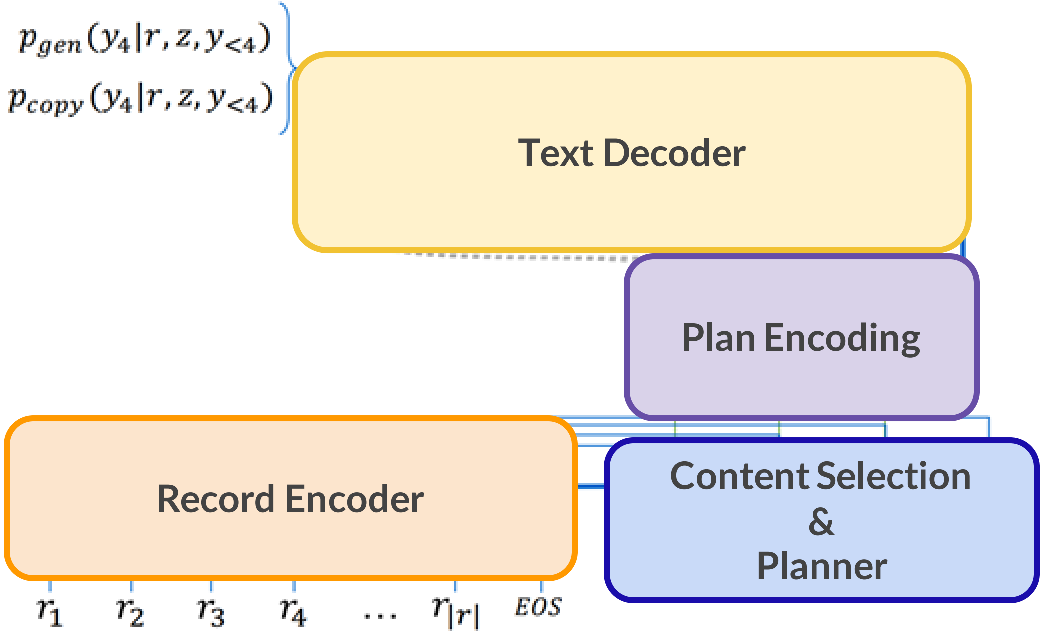
✗ **Computationally expensive**
Both for finetuning and serving these models a lot of computational resources are needed.

Table-to-Text Key Systems

1. NCP (2018)
2. DATA-TRANS (2019)
3. TableGPT (2020)
4. T5* (2020)
5. Plan-then-Generate (2021)

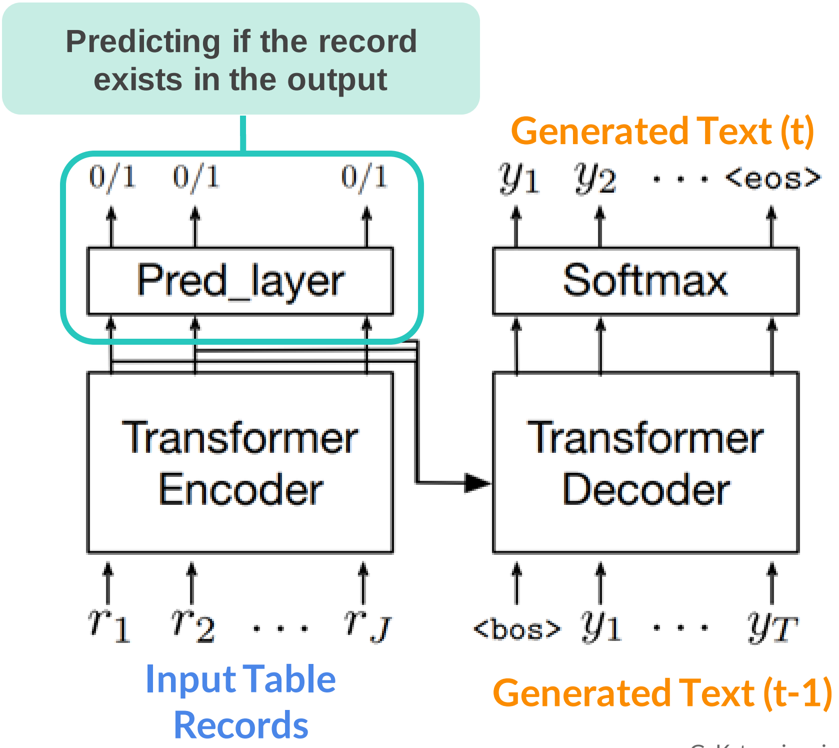
* Take T5 as is and finetune-evaluate on ToTTo

NCP (Neural Content Planning)



- The separation between stages allows for **easier debugging and understanding** of the model
- **They create their own embeddings from scratch**
    - Useful if your dataset is of a specific domain (eg. ROTOWIRE in their case)

DATA-TRANS (Data-to-Text Transformer)



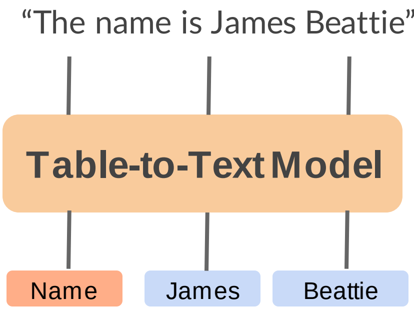
➔ **Target Prediction Auxiliary Task**
The encoder is forced to learn meaningful representations by using the target prediction auxiliary task.

➔ **ROTOWIRE Augmentation**
They augment the ROTOWIRE dataset by changing record values without altering the final game outcome.

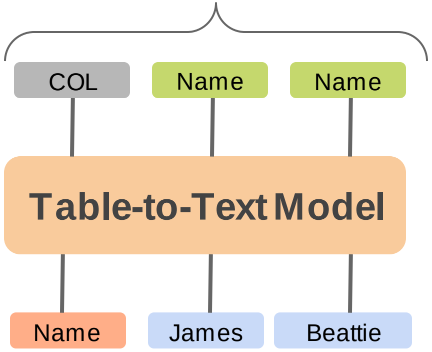
➔ **Training from scratch**
They train the transformer model from scratch, not utilizing any existing pretrained LMs.

TableGPT

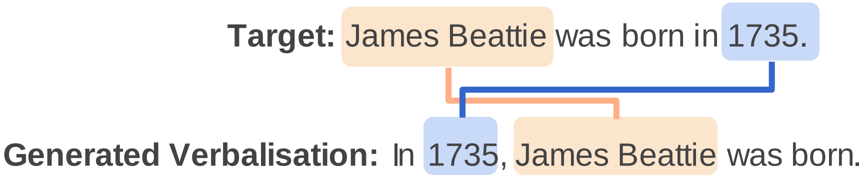
1. Language Model Loss L_{LM}



2. Structure Reconstruction Loss L_{SR}



3. Content Ordering Loss L_{CO}



Their training loss consists of 3 components

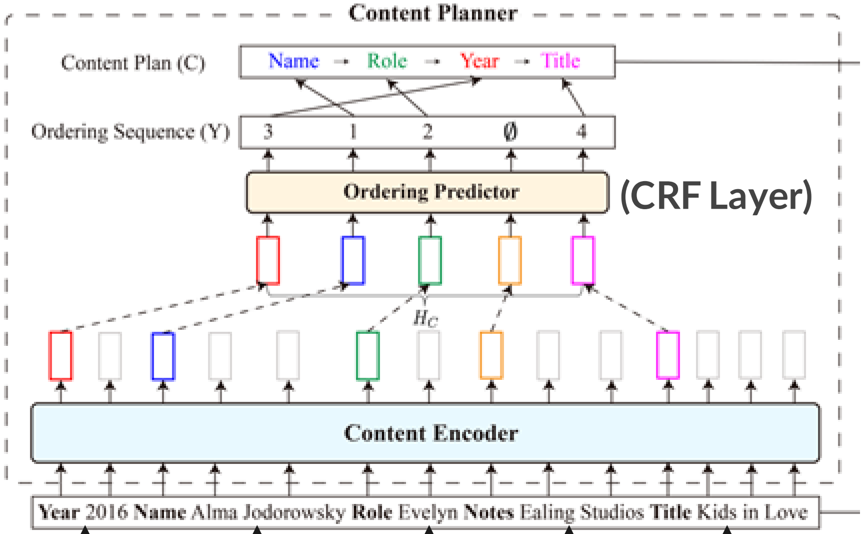
1. **Language Model Loss**
   The model learns how to verbalise tables.

2. **Structure Reconstruction Loss**
   The model learns how to embed structural information.

3. **Content Ordering Loss**
   Promote generation of high fidelity text.

TableGPT performs great on the few-shot setting (50-500 training prompts).

Plan-then-Generate



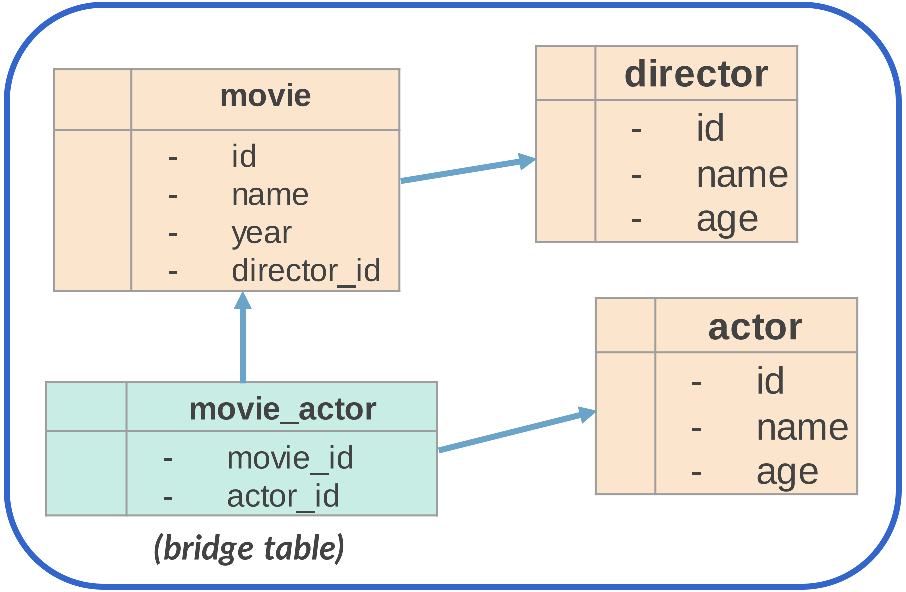
Year	Name	Role	Notes	Title
2016	Alma Jodorowsky	Evelyn	Ealing Studios	Kids in Love

- ➔ They use **BERT as their content encoder.**
  - ◆ Much needed approach since we have moved to the **ToTTo dataset which has great textual diversity.**
- ➔ The ordering is then defined by a CRF layer (content planner).
- ➔ They use BART to generate the final text based on the table and the generated plan.
- ➔ To teach the model to remain faithful to the plan they employ RL-training.

The Graph-to-Text Problem

Graph-to-Text

Given a graph generate text that expresses the information of the whole graph or parts of it.



"The database has information about the ***movie domain****. For each movie it contains its* ***name*** *and* ***release year*** *along with the* ***directors*** *and* ***actors*** *that participated."*

Datasets

Year	Dataset	Type/Domain	Examples
2017-20	WebNLG (v3)	DBPedia	16,905
2017-20	LDC2020	Who did what to whom?	59,255
2020	AGENDA	Knowledge Graph	40,720
2020	LOGIC2TEXT	Wikipedia	10,753
2020	WITA	Wikipedia	55,400
2020	GenWiki	DBPedia	1.3mil
2020	ENT-DESC	Knowledge Graphs	110,000
2021	WikiGraphs	Wikipedia	23,522
2021	Map2Seq	OpenStreetMap	7,772
2021	DART	Wikipedia+Restaurant	82,191
2021	EventNarrative	EventKG+Wikidata	224,428

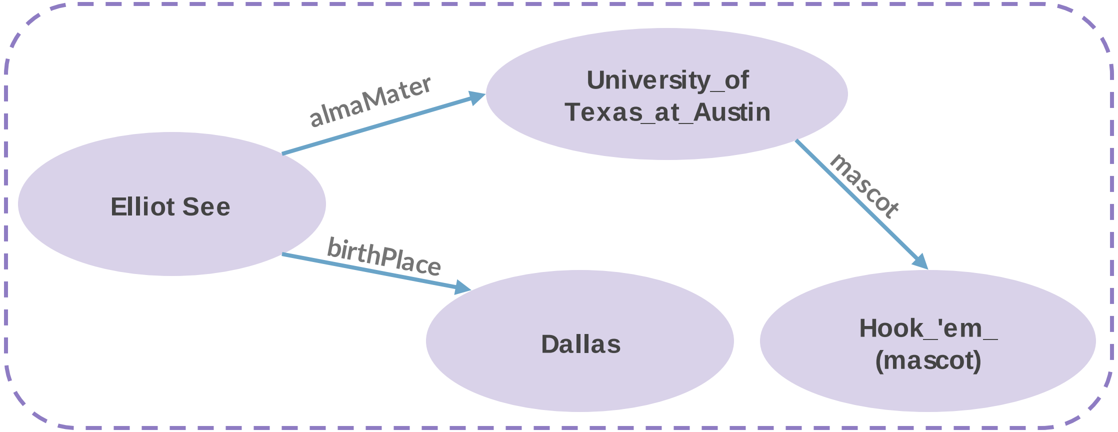
Most influential datasets that we analyze next.

Web NLG - 2020

➔ **Size:** 17K triplets

Comprises of sets of triplets describing facts (entities and relations between them) and the corresponding facts in form of natural language text.

The data is a set of triples extracted from **DBpedia** and the text is a verbalisation of these triples.



"Elliot See was born in Dallas, and graduated from the University of Texas at Austin where Hook 'em is the mascot."

Still getting updated every ~3 years

LDC - 2020

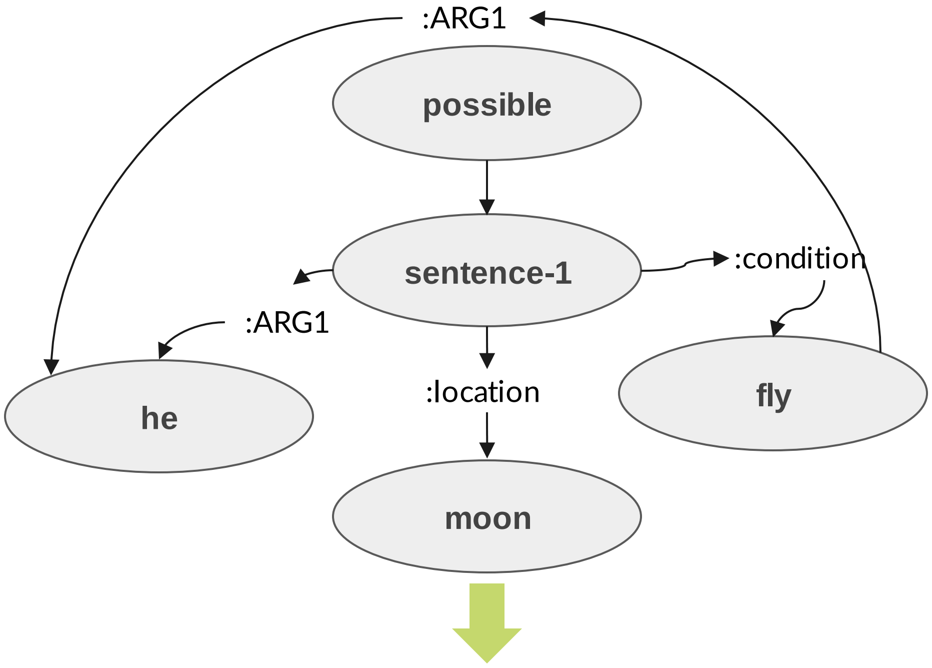
A dataset facilitating the **Abstract Meaning Representation (AMR) to text task**.

AMR captures "who is doing what to whom" in a sentence. Each sentence is paired with a graph that represents its meaning in a tree-structure.

- **Data sources:**
  - Forum discussions
  - Journals
  - Blogs
  - News texts

Still getting updated every ~3 years

- **Size:** 59K triplets



"If he flies he could go to the moon."

Map2Seq- 2021

Human written natural language navigation instructions for routes in OpenStreetMap with a focus on visual landmarks.

Original route

Graph representation

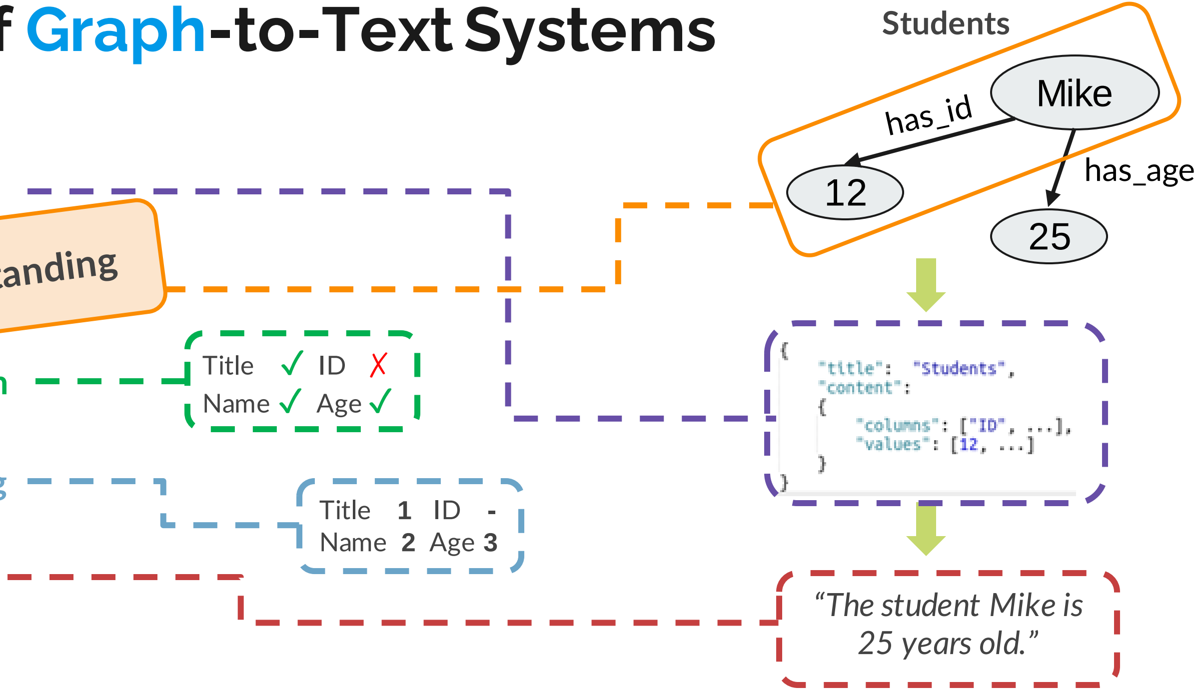
"Go to the lights and turn right. Go to the second set of lights and turn left. Tompkins Square park will be on your right. Go about half way down the block. Stop at Avant Garden on your right."

➔ **Dataset size:** 7,672 navigation instructions

➔ All of the instructions were manually validated for their correctness.

Challenges of Graph-to-Text Systems

- → Representation
- → Graph Understanding
- → Content selection
- → Content planning
- → Text generation

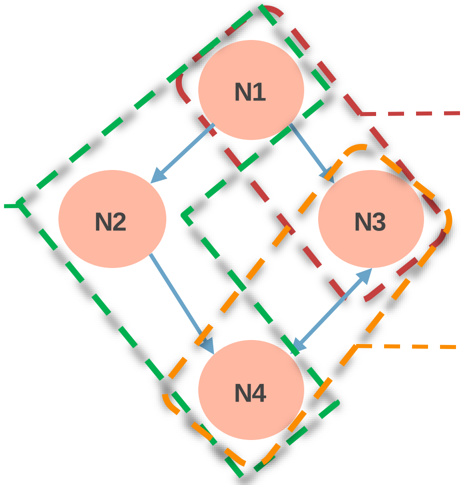


These table-to-text solutions can be applied directly (or with slight modifications) to Graph-to-Text too.

Unique Challenge of Graph-to-Text

Challenge: Encoding the graph structure and the **information we get from it** into a meaningful representation

How do you encode that N1 is connected to N4 through N2?



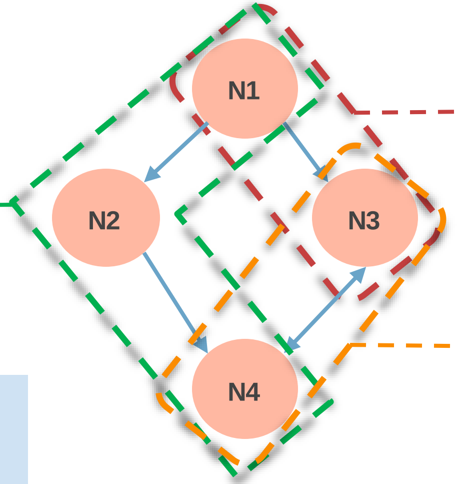
How do you encode an edge connection between N1 and N3?

How do you encode the bidirectionality of N3 and N4?

Unique Challenge of Graph-to-Text

Challenge: Encoding the graph structure and the **information we get from it** into a meaningful representation

How do you encode that N1 is connected to N4 through N2?



How do you encode an edge connection between N1 and N3?

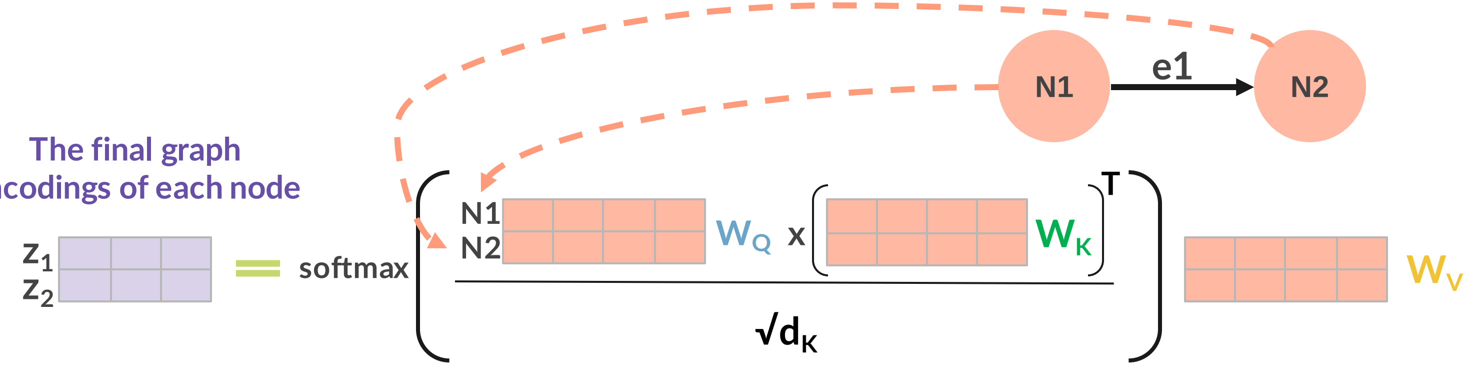
How do you encode the bidirectionality of N3 and N4?

By solving the graph encoding challenge we can generalize our Graph-to-Text task to a Data-to-Text one, which in turn allows the usage of Table-to-Text solutions.

Encoding: Enhancing Self-Attention

The transformer self-attention mechanism was proposed initially for text-to-text problems, meaning that it expects a **sequence of tokens.**

The final graph encodings of each node

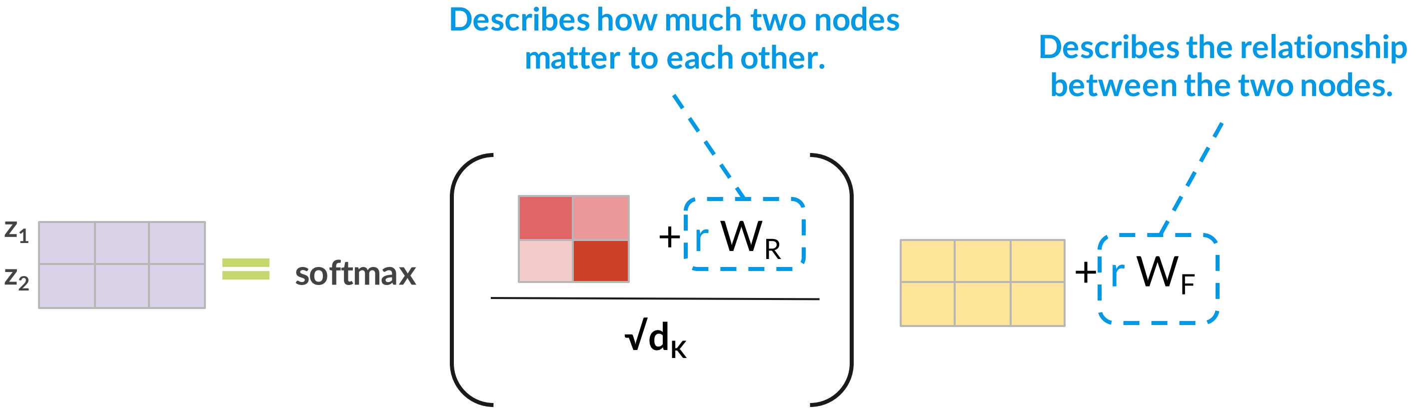


The mechanism does not utilize the explicit relationships (edges) that may exist between the two nodes.

[66] Graph Transformer for Graph-to-Sequence Learning (2019)
[69] Modeling Graph Structure in Transformer for Better AMR-to-Text Generation (2019)

Encoding: Enhancing Self-Attention

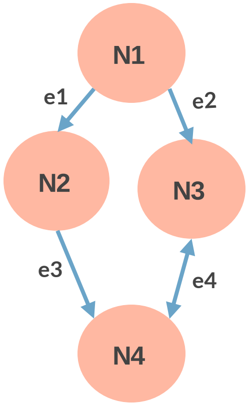
Solution: Enhance the attention mechanism by including an encoding of the relationships between nodes



But how can we generate r which describes the relationship between every node combination?

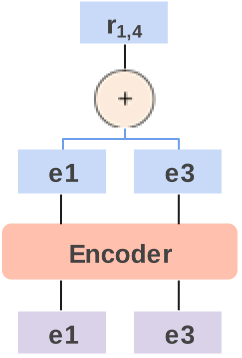
[66] Graph Transformer for Graph-to-Sequence Learning (2019)
[69] Modeling Graph Structure in Transformer for Better AMR-to-Text Generation (2019)

Encoding: Enhancing Self-Attention



		to			
		N1	N2	N3	N4
from	N1	-	e1	e2	e1,e3
	N2	NO	-	e3, e4	e3
	N3	NO	NO	-	e4
	N4	NO	NO	e4	-

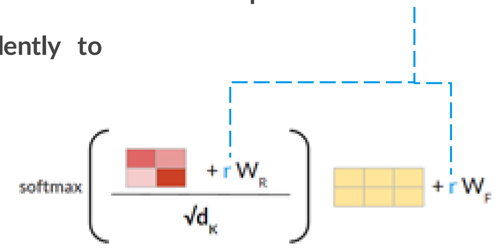
Shortest paths between every node



Encode each path independently to get an embedding

$r_{1,1}$	$r_{1,2}$	$r_{1,3}$	$r_{1,4}$
$r_{2,1}$	$r_{2,2}$	$r_{2,3}$	$r_{2,4}$
$r_{3,1}$	$r_{3,2}$	$r_{3,3}$	$r_{3,4}$
$r_{4,1}$	$r_{4,2}$	$r_{4,3}$	$r_{4,4}$

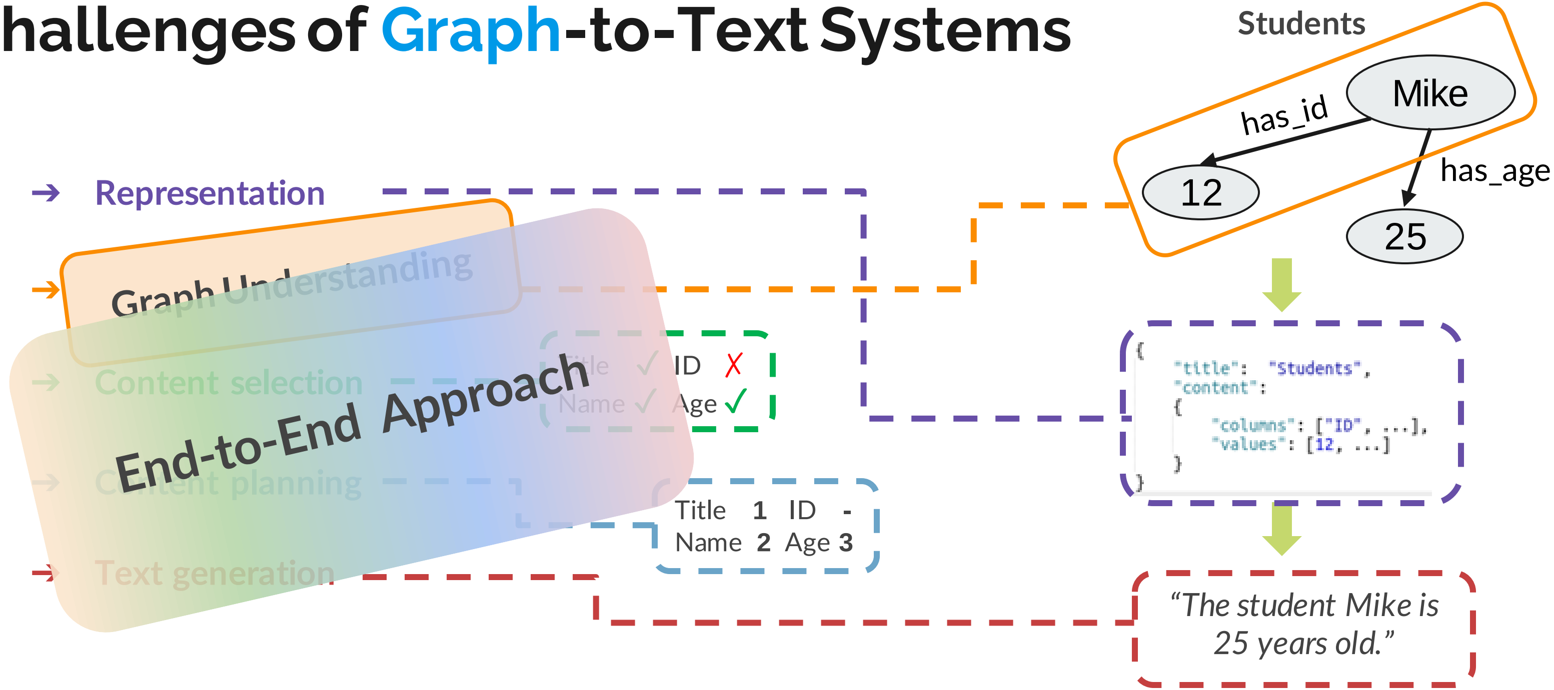
Our final relationship representation r



There are other methods to encode each path:

- SUM of embeddings
- AVG of embeddings
- 1D Convolution
- RNNs

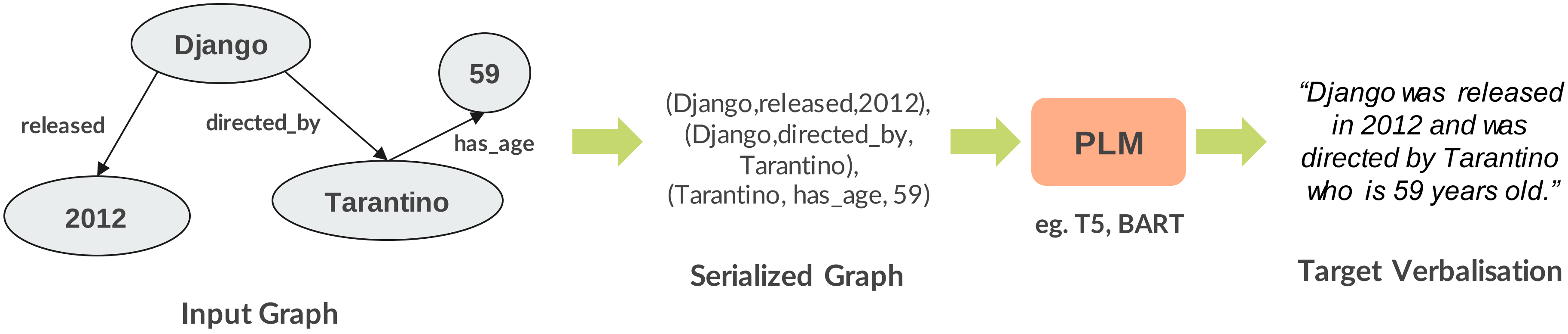
Challenges of Graph-to-Text Systems



These table-to-text solutions can be applied directly (or with slight modifications) to Graph-to-Text too.

End-to-End Solution

As in Table-to-Text we can simply define a way of serializing our graph to text and then simply feed it to a pretrained LM.

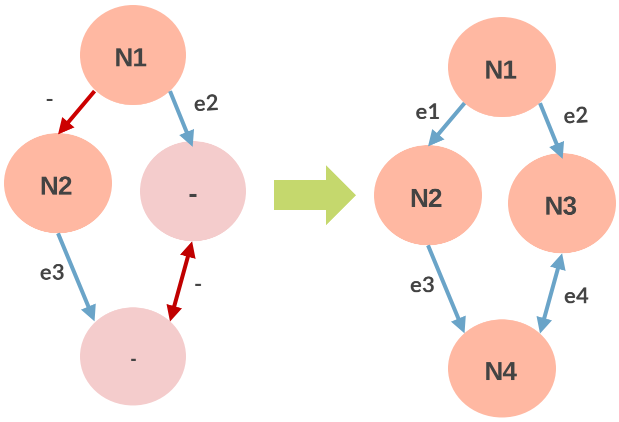


Assumption: The model will be able to catch and embed the relationships that exist between nodes.

Can we help the model to have a better understanding of what a graph is?

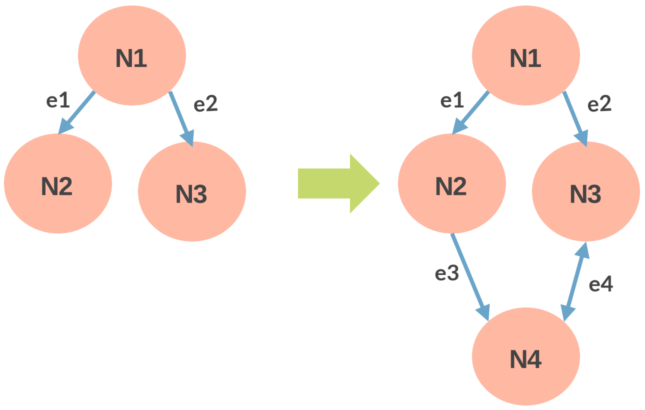
Graph Understanding: Pretraining Tasks

The pretraining tasks aim at improving the graph awareness of PLMs.



Node and edge denoising

Goal: Capturing knowledge about node and edge values.



Sub-graph denoising

Goal: Enforces the model to predict a sub-graph, thus facilitating the graph-level learning.

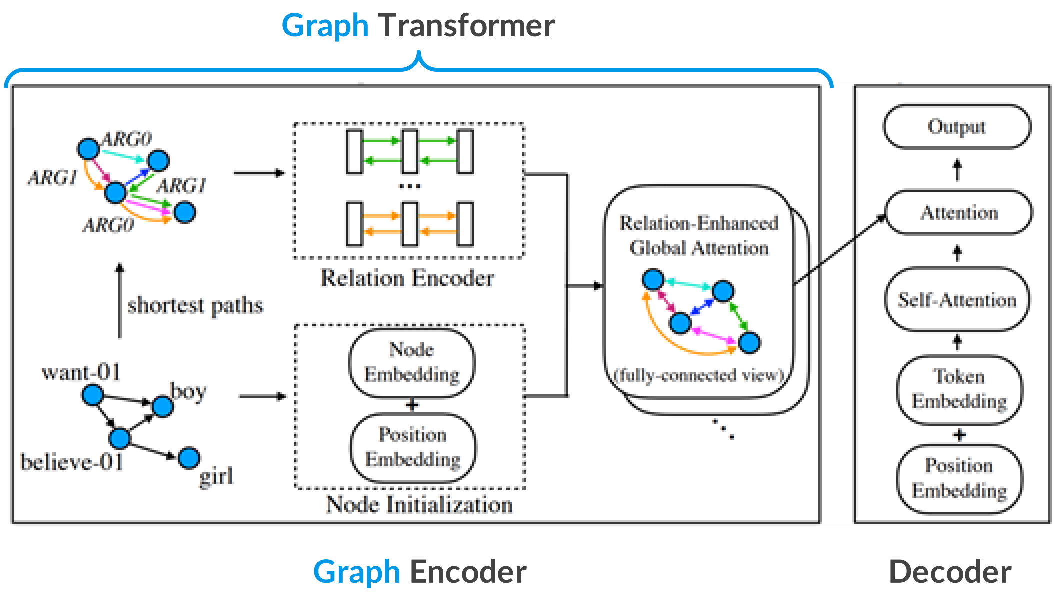
Graph-to-Text Key Systems

1. Graph Transformer (2019)
2. T5* (2020)
3. AMRBART (2022)

* Take T5 as is and finetune-evaluate on WebNLG and LDC datasets.

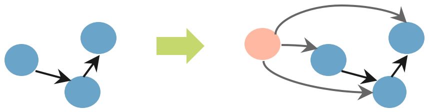
By no means this list is exhaustive, it is just a sample we consider sufficient for the scope of the tutorial.

Graph Transformer



➔ **Path Encoding**
They use bidirectional GRUs for encoding each path.

➔ **Graph Global Context**
They introduce a virtual node that has direct edges to all other nodes, which will capture global graph information.

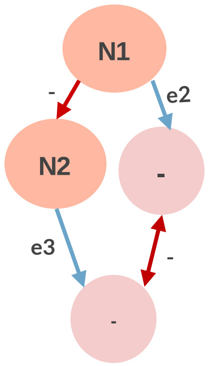


➔ **Copy Mechanism**
For the OOV problem they also use a copy mechanism exactly like in table-to-text

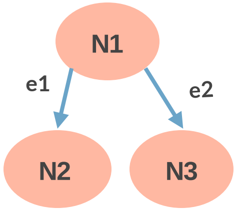
AMRBART

First pretrain BART on Graph structure understanding with tasks aiming at:

1. Capturing node/edge knowledge
2. Comprehend the meaning of a graph



Node and edge denoising



Sub-graph denoising

They also propose a new pretraining and finetuning setting that improves performance, called **unified framework**.

Non-unified Pretraining

<g> g_1, g_2, [mask], …, g_m </g> → <g> g_1, g_2, g_3, …, g_m </g>

Unified Pretraining

<x> x_1, x_2, [mask] …, x_n</x> <g> g_1, g_2, [mask], …, g_m </g>

↓

<x> x_1, x_2, x_3 …, x_n</x> <g> g_1, g_2, g_3, …, g_m </g>

The unified framework helps the model to better learn the interaction between textual and AMR information during pre-training.

Data-to-Text Evaluation

Evaluation Metrics

In Data-to-Text, evaluating the performance of a model automatically means quantifying how similar two verbalisations are.

"The average project has a duration of 3 years."	**vs.**	"A project, in average, has a duration of 3 years."
"Katerina is 25 years old."	**vs.**	"Chris is 26 years old."

Heuristic

Table-to-Text specific

- Relation Generation
- Content Selection
- Content Ordering

n-gram matching

- BLEU
- METEOR
- **PARENT (table-to-text)**
- ROUGE
- CHRF++

Semantic

- BERTscore
- BLEURT
- MoverScore
- Embedding Distance

Evaluation Metrics - PARENT

A table-to-text specific metric that is based on BLEU but also takes into account the contents of the table.

Reference:	Michael Dahlquist (December 22 , 1965 -- July 14 , 2005) was a drummer in the Seattle band Silkworm .	***BLEU***	***PARENT***
Candidate 1:	Michael Dahlquist (December 22 , 1965 -- July 14 , 2005) was a drummer in the California band Grateful Dead .	**0.79**	0.76
Candidate 2:	Michael Dahlquist (December 22 , 1965 -- July 14 , 2005) was a drummer .	0.71	0.82
Candidate 3:	Michael Dahlquist (December 22 , 1965 -- July 14 , 2005) was a drummer from Seattle, Washington .	0.73	**0.84**

Michael Dahlquist	
Birth name	Michael Dahlquist
Born	December 22, 1965 Seattle, Washington, US
Died	July 14, 2005 (aged 39) Skokie, Illinois, US
Genres	Indie rock
Occupation(s)	Musician
Instrument(s)	Drums
Years active	1990–2005

✓ Does not punish the model unfairly ✓ Easily interpretable ✗ Fails to understand semantic similarities

Results

	Model	Dataset	BLEU	PARENT
Table-to-Text	NCP	ROTOWIRE	16.19	-
	DATA-TRANS	ROTOWIRE	**19.97**	-
	T5	ToTTo	**49.5**	58.4
	Plan-then-Generate	ToTTo	49.2	**58.7**
Graph-to-Text	Graph Transformer	LDC2017	29.8	-
	T5	LDC2017	45.8	-
	AMRBART	LDC2017	49.8	-

➔ A lot of room for improvement for the challenging ROTOWIRE dataset.

➔ Intricate solution for ToTTo did not manage to beat by a lot the simple T5 application.

➔ But, pretraining for graph understanding in AMRBART achieved significant improvements.

Challenges and Research Opportunities in Data-to-Text

Purpose may not be clear

In an NLIDB the user has a purpose stated by the NL query.
Eg. *How good is the cheapest laptop?*

Many ways to verbalise a table or graph, especially the bigger they are.

Model	CPU	Display	Price
Laptop 1	i3	17''	1050
Laptop 2	i7	15''	950

Laptop 1 has an i3 CPU, a display of 17'' and is priced at 1050. Laptop 2 has an i7 CPU, a display of 15'' and a price of 950.

Laptop 2 is the cheapest laptop with the fastest CPU but the smallest display.

The most expensive laptop is Laptop 1.

What about the opposite?

Laptop 1 has an i3 CPU, a display of 17'' and is priced at 1050. Laptop 2 has an i7 CPU, a display of 15'' and a price of 950.

Model	CPU	Display	Price
Laptop 1	i3	17''	1050
Laptop 2	i7	15''	950

Given a text manage to organize its contents into a structured format.

Why is this useful?

- Tables or any other structured format can be easily parsed by machines
- Automatic metadata generation
- New datasources for data analysis and exploration

Utilizing or Creating Data-to-Text PLMs

So far, the state of the art is achieved by using text-to-text PLMs.

But, the Data-to-Text task differs from text-to-text suggesting the possibility of creating PLMs specific for our use case.

Requirements

- Appropriate architecture (not necessary to follow the transformer solution as is)
- Sufficient amount of silver data:
  - Table understanding: Web Table Corpus (WDC), 233 million tables
  - Graph understanding: DBPedia, 228 million entities
- Compute and expertise

Such attempt was made by TaBERT creating a BERT variation pretrained for table understanding.

No Data-to-Text dataset on the DB domain

Why don't just use existing datasets for table-to-text?

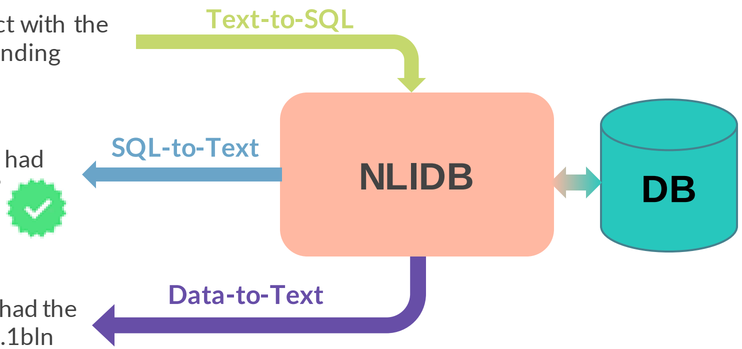
- We can utilize extra schema information
- A database will not follow conventions of a Wikipedia table
- A query expects something specific as an answer, not a general verbalisation

Bringing it all together

NLIDB - The big challenge

Goal: Combining all these different domains into one system that can be considered a **Natural Language Interface of Databases (NLIDB)**.

1. Find the project with the biggest funding
2. What is the project that had the largest funding?
3. The project NEW-EU had the biggest funding of 2.1bln



The simple solution:

1. Get a well performing model from each field.
2. Train them on their respective datasets.
3. The rest is a technical challenge of how to serve these 3 models.

But many challenges arise.

Challenges: Database Generalizability

For a system to be useful it must be able to work correctly on databases that no training data exist.
Current datasets (eg. Spider) are of high quality but are not able to cover the diversity and difficulties of real-world databases.

Databases are used in every domain from life sciences to e-commerce

Database names and columns might not be “PLM friendly” eg. *usr*

Production databases tend to be much bigger than the ones in existing datasets
(eg. Spider databases have 4 tables on average)

Possible directions

- Creating new datasets or expanding current ones
- Few-shot training sets to bootstrap the models
- Pretraining tasks and datasets that focus on generalizability
- Query or DB preprocessing

Challenges

Error Propagation

For a user interaction with the NLIDB to be considered successful all 3 components must work correctly.

Performance analysis and evaluation becomes harder since we need a common Benchmark for all the components.

Latency

All of our components' best solutions utilize PLMs (mostly T5).

$T5_{\text{Text-to-SQL}}$ $T5_{\text{SQL-to-Text}}$ $T5_{\text{Data-to-Text}}$

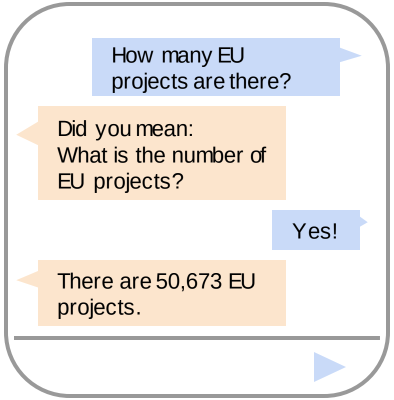
T5 variations (10^7 - 10^9 parameters) have a significant latency when producing inferences. The **latency is 3x** since all the components use T5 or similar PLMs.

NLIDB solutions must address this issue by focusing on:

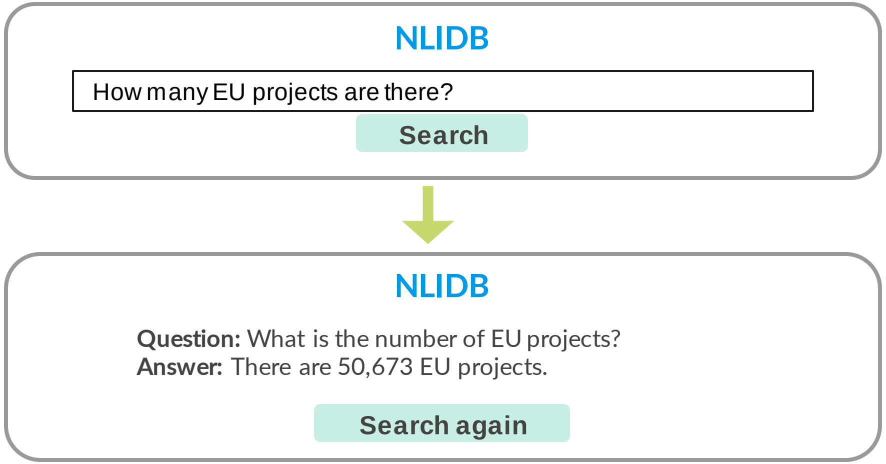
- **Efficiency**
- **Model size**

Challenges: User Interface

Designing an intuitive and easy to understand interface, which will not overwhelm the user.



Conversational Approach



“Search Engine” Approach

[73] Cluster-Driven Navigation of the Query Space (2016)
[74] Overview of Data Exploration Techniques (2015)

Challenges: Incorporating other fields

In this tutorial we explored the 3 main components of a NLIDB. However, there are more fields and by incorporating them we can improve the user experience.

Query Recommendation

Query recommendations aim at aiding the user build a query that cover their needs based on previous interactions or data insights.

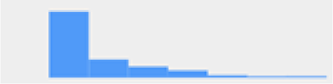
How many projects started in 2019?

How many projects started after 2019?

How many projects started in 2019 from a French organization?

Data Exploration

Data exploration addresses query results that have a massive number of rows they are difficult to interpret and end up overwhelming the user.



"In month May of 2021 more projects started compared to month May of the past 10 years."

Median/Avg/Max/Min

Demo

DatAgent

https://bit.ly/datagent

Not mobile friendly 💻

DatAgent is a smart data assistant, developed by the DARELAB team, which works as a NLIDB, integrating solutions for

✓ **Text-to-SQL** ✓ **SQL-to-Text** ✓ **Data-to-Text**

✓ Query Recommendations ✓ Data Exploration

The online version by default runs on the CORDIS database:

A real-world production database used by the European Commission to store information about EU-funded programs such as projects, participants, institutions, etc.

Some example queries that succeed

- Find the number of projects that started in 2015
- Which are the institutions in France?
- Find the projects of the institutions in France *(requires 4 JOINs)*

But still a lot of work needs to be done!

- Which project had the **biggest** cost?
- When is the end date of project with acronym ALFRED?

Noticed something interesting or you have a question?
Feel free to talk to us or contact us "offline".

Thank you! Questions?

References (1/13)

[1] E. F. Codd. **Seven Steps to Rendezvous with the Casual User**. 1974. IFIP Working Conference Data Base Management.

[2] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. **Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning.** CoRR, September 2017

[3] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev. 2018. **Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task**. EMNLP 2018.

[4] G. Katsogiannis-Meimarakis, G. Koutrika. 2023. **A survey on deep learning approaches for text-to-SQL**. The VLDB Journal .

[5] U. Brunner and K. Stockinger. 2020. **ValueNet: A Neural Text-to-SQL Architecture Incorporating Values**.

[6] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J. Lou, T. Liu, and D. Zhang. 2019. **Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation.** Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

References (2/13)

[8] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Ł. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes and J. Dean. 2016. **Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation**.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. 2017. **Attention Is All You Need**. NIPS 2017.

[10] D. Jacob, C. Ming-Wei, L. Kenton and T. Kristina. 2019. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 4171–4186.

[11] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov. 2019. **RoBERTa: A Robustly Optimized BERT Pretraining Approach**.

[12] P. Yin, G. Neubig, W. Yih and S. Riedel. 2020. **TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data** Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

References (3/13)

[13] T. Yu, C. Wu, X. V. Lin, B. Wang, Y. C. Tan, X. Yang, D. Radev, R. Socher and C. Xiong. 2020. **GraPPa: Grammar-Augmented Pre-Training for Table Semantic Parsing**.

[14] S. Hochreiter and J. Schmidhuber . 1997. **Long Short-term Memory**. Neural computation. 9. 1735-80.

[15] T. Mikolov, K. Chen, G. Corrado and J. Dean. 2013. **Efficient Estimation of Word Representations in Vector Space**.

[16] J. Pennington, R. Socher and C. D. Manning. 2014. **GloVe: Global Vectors for Word Representation**. EMNLP 2014.

[17] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer. 2020. **BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension**. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics

[18] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu. 2020. **Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer**. Journal of Machine Learning Research

References (4/13)

[19] L .Dong and M. Lapata. 2016. **Language to Logical Form with Neural Attention**. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).

[20] X. V. Lin, R. Socher and C. Xiong. **Bridging Textual and Tabular Data for Cross-Domain Text-to-SQL Semantic Parsing**. Findings of the Association for Computational Linguistics: EMNLP 2020.

[21] T. Scholak, N. Schucher, D. Bahdanau. 2021. **PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models**. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing

[22] X. Xu, C. Liu and D. Song. 2017. **SQLNet: Generating Structured Queries From Natural Language Without Reinforcement Learning**.

[23] W. Hwang, J. Yim, S. Park and M. Seo. 2019. **A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization**.

[24] Q. Lyu, K. Chakrabarti, S. Hathi, S. Kundu, J. Zhang and Z. Chen. 2020. **Hybrid Ranking Network for Text-to-SQL**.

References (5/13)

[25] T. Shi, K. Tatwawadi, K. Chakrabarti, Y. Mao, O. Polozov, and W. Chen. 2018. **IncSQL: Training Incremental Text-to-SQL Parsers with Non-Deterministic Oracles**.

[26] B. Wang, R. Shin, X. Liu, O. Polozov, M. Richardson. 2020. **RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers**. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[27] K. Xuan, Y. Wang, Y. Wang, Z. Wen, Y. Dong. 2021. **SeaD: End-to-end Text-to-SQL Generation with Schema-aware Denoising**.

[28] L. Zhao, H. Cao, Y. Zhao. 2021. **GP: Context-free Grammar Pre-training for Text-to-SQL Parsers**.

[29] C. Wang, K. Tatwawadi, M. Brockschmidt, P. Huang, Y. Mao, O. Polozov and R. Singh Robust. 2018. **Text-to-SQL Generation with Execution-Guided Decoding**.

[30] B. Bogin, M. Gardner, J. Berant. 2019. **Global Reasoning over Database Structures for Text-to-SQL Parsing**. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing

References (6/13)

[31] G. Koutrika, A. Simitsis, and Y. E. Ioannidis. 2010. **Explaining structured queries in natural language**. IEEE 26th International Conference on Data Engineering.

[32] D. Guo, Y. Sun, D. Tang, N. Duan, J. Yin, H. Chi, J. Cao, P. Chen, and M. Zhou. 2018. **Question generation from SQL queries improves neural semantic parsing**. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.

[33] K. Xu, L. Wu, Z. Wang, Y. Feng, and V. Sheinin. 2018. **SQL-to-text generation with graph-to-sequence model**. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.

[34] D. Ma, X. Chen, R. Cao, Z. Chen, L. Chen, and K. Yu. 2022. **Relation-aware graph transformer for sql-to-text generation**. Applied Sciences.

[35] K. Wu, L. Wang, Z. Li, A. Zhang, X. Xiao, H. Wu, M. Zhang, and H. Wang. 2021. **Data augmentation with hierarchical SQL-to-question generation for cross-domain text-to-SQL parsing**. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.

References (7/13)

[36] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, and M. Zhou. 2020. **CodeBERT: A Pre-Trained Model for Programming and Natural Languages**. In Findings of the Association for Computational Linguistics: EMNLP 2020.

[37] W. Ahmad, S. Chakraborty, B. Ray, and K.W. Chang. 2021. **Unified Pre-training for Program Understanding and Generation**. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics.

[38] Y. Wang, W. Wang, S. Joty, and S. C.H. Hoi. 2021. **CodeT5: Identifier-aware Unified Pre-trained Encoder-Decoder Models for Code Understanding and Generation**. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.

[39] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. **Bleu: a method for automatic evaluation of machine translation**. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics

[40] M. Popovic. 2015. **chrF: character n-gram F-score for automatic MT evaluation**. In Proceedings of the Tenth Workshop on Statistical Machine Translation.

[41] S. Banerjee and A. Lavie. 2005. **METEOR: An automatic metric for MT evaluation with improved correlation with human judgments**. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization

References (8/13)

[42] T. Sellam, D. Das, and A. Parikh. 2020. **BLEURT: Learning Robust Metrics for Text Generation**. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[43] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. 2020. **BERTScore: Evaluating Text Generation with BERT**. In Eighth International Conference on Learning Representations.

[44] B. Ell, D. Vrandečić, E. Simperl. 2015. **SPARTIQULATION: Verbalizing SPARQL Queries**. The Semantic Web: ESWC 2012 Satellite Events.

[45] P. Yin and G. Neubig. 2017. **A Syntactic Neural Model for General-Purpose Code Generation**. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).

[46] V. Hristidis and Y. Papakonstantinou. 2002. **Discover: Keyword Search in Relational Databases**. In VLDB.

[47] V. Hristidis, L. Gravano, and Y. Papakonstantinou. 2003. **Efficient IR-style Keyword Search over Relational Databases**. In VLDB.

[48] Y. Luo, X. Lin, W. Wang, and X. Zhou. 2007. **Spark: Top-k Keyword Query in Relational Databases**. In ACM SIGMOD.

References (9/13)

[49] Z. Zeng, M. L. Lee, and T. Wang Ling. 2016. **Answering Keyword Queries involving Aggregates and GROUPBY on Relational Databases**. EDBT.

[50] L. Blunschi, C. Jossen, D. Kossmann, M. Mori, and K. Stockinger. 2012. **SODA: Generating SQL for Business Users**. PVLDB 5, 10 (2012), 932–943.

[51] F. Li and H. V. Jagadish. 2014. **Constructing an Interactive Natural Language Interface for Relational Databases**. PVLDB 8, 1 (Sept. 2014), 73–84.

[52] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Özcan. 2016. **ATHENA: An Ontology-Driven System for Natural Language Querying over Relational Data Stores**. VLDB.

[53] Rémi Lebret, David Grangier, and Michael Auli. 2016. **Neural Text Generation from Structured Data with Application to the Biography Domain**. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing

[54] Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. **Challenges in Data-to-Document Generation**. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

References (10/13)

[55] Craig Thomson, Ehud Reiter, and Somayajulu Sripada. 2020. **SportSett:Basketball - A robust and maintainable data-set for Natural Language Generation**. In Proceedings of the Workshop on Intelligent Information Processing and Natural Language Generation

[56] Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. **ToTTo: A Controlled Table-To-Text Generation Dataset**. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)

[57] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. **Table-to-text generation by structure-aware seq2seq learning**. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18)

[58] Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. **TableGPT: Few-shot Table-to-Text Generation with Table Structure Reconstruction and Content Matching.** In Proceedings of the 28th International Conference on Computational Linguistics, pages 1978–1988, Barcelona, Spain (Online). International Committee on Computational Linguistics. (COLING)

[59] Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. **Plan-then-Generate: Controlled Data-to-Text Generation via Planning.** In Findings of the Association for Computational Linguistics: EMNLP 2021

[60] Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. **Data-to-text generation with content selection and planning.** In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19)

References (11/13)

[61] Heng Gong, Wei Bi, Xiaocheng Feng, Bing Qin, Xiaojiang Liu, and Ting Liu. 2020. **Enhancing Content Planning for Table-to-Text Generation with Data Understanding and Verification**. In Findings of the Association for Computational Linguistics: EMNLP 2020

[62] Li Gong, Josep Crego, and Jean Senellart. 2019. **Enhanced Transformer Model for Data-to-Text Generation.** In Proceedings of the 3rd Workshop on Neural Generation and Translation, pages 148–156, Hong Kong. Association for Computational Linguistics.

[63] Bao, Junwei, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. **Table-to-text: Describing table region with natural language.** In Proceedings of the AAAI conference on artificial intelligence, vol. 32, no. 1. 2018.

[64] Mihir Kale and Abhinav Rastogi. 2020. **Text-to-Text Pre-Training for Data-to-Text Tasks**. In Proceedings of the 13th International Conference on Natural Language Generation, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

[65] Raphael Schumann and Stefan Riezler. 2021. **Generating Landmark Navigation Instructions from Maps as a Graph-to-Text Problem.** In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 489–502, Online. Association for Computational Linguistics.

[66] Cai, Deng, and Wai Lam. **Graph transformer for graph-to-sequence learning**. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 05, pp. 7464-7471. 2020.

References (12/13)

[67] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. **Creating Training Corpora for NLG Micro-Planners.** In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

[68] Li Gong, Josep Crego, and Jean Senellart. 2019. **Enhanced Transformer Model for Data-to-Text Generation.** In Proceedings of the 3rd Workshop on Neural Generation and Translation, pages 148–156, Hong Kong. Association for Computational Linguistics.

[69] Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. **Modeling Graph Structure in Transformer for Better AMR-to-Text Generation.** In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)

[70] Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. **Investigating Pretrained Language Models for Graph-to-Text Generation**. In Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI, pages 211–227, Online. Association for Computational Linguistics.

[71] Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. **Graph Pre-training for AMR Parsing and Generation.** In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

[72] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. **TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data.** In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8413–8426, Online. Association for Computational Linguistics.

References (13/13)

[73] Sellam, Thibault, and Martin Kersten. **Cluster-driven navigation of the query space.** IEEE Transactions on Knowledge and Data Engineering 28, no. 5 (2016): 1118-1131.

[74] Idreos, Stratos, Olga Papaemmanouil, and Surajit Chaudhuri. **Overview of data exploration techniques.** In Proceedings of the 2015 ACM SIGMOD international conference on management of data, pp. 277-281. 2015.

[75] Dhingra, Bhuwan, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W. Cohen. **Handling divergent reference texts when evaluating table-to-text generation.** arXiv preprint arXiv:1906.01081 (2019).