Logical Table-to-Text Generation: Challenges, Methods, and Reasoning

Lena Trigg and Dean Hougen
School of Computer Science, University of Oklahoma
Norman, OK, USA
lena.trigg@ou.edu, hougen@ou.edu

Abstract

Logical Table-to-Text (LT2T) generation requires models to both verbalize tabular data and reason over it - performing comparisons, aggregations, and causal inference. While many generation tasks struggle with similar analytical demands, LT2T provides a structured perspective on reasoning capabilities in natural language generation. This survey uses LT2T as a lens to focus on reasoning in data-to-text tasks. By focusing narrowly on LT2T, we present a deep taxonomy of methods that inject, structure, or verify reasoning steps, allowing a level of technical granularity missing in broader surveys. We review representative models and evaluation metrics, and highlight how LT2T techniques transfer to general generation challenges involving logic, numeracy, and faithfulness. Our goal is to distill lessons from LT2T that apply more widely, while also guiding future research in table-based reasoning.

1 Introduction

Tabular data is pervasive in domains such as finance, science, and sports, but is often obscure to non-experts. *Table-to-Text* (T2T) models bridge this gap by transforming structured tables into readable summaries. While early neural T2T models improved fluency, they struggled to meet users' analytical expectations, often producing surface-level descriptions that merely repeat table content.

Logical Table-to-Text (LT2T) raises the bar by requiring reasoning-based generation. Instead of stating individual facts, LT2T models must infer logical relationships. Figure 1 contrasts shallow and logical outputs to illustrate this difference. Generating logical-level output introduces challenges such as logical fidelity, numerical accuracy, and controllable content selection, which also affect many other natural language generation (NLG) tasks.

We chose LT2T to survey as it offers a compact, well-defined testbed for the broader NLP goal of trustworthy text generation with reasoning because the input structure is explicit and every valid operation is enumerable. At the same time, limiting the scope to LT2T allows us to offer a deep, actionable taxonomy of reasoning strategies that would be too shallow or generic in a broader survey. We categorize methods by how they address issues to enhance reasoning. This is especially useful for practitioners entering the field or designing systems that require interpretable, faithful reasoning.

Statistics of Three Countries in Middle East			
Country	**Area (km²)**	**Population**	**Density (per km²)**
Israel	21,937	10,100,000	460
Iraq	438,317	45,521,000	104
Iran	1,648,195	87,500,000	53
Surface-level Generation			
Sentence: Israel has an area of 21,937 km² and on average 460 persons live in each km².			
Logical-level Generation			
Sentence: Israel has the smallest area among the three countries with the highest population per square kilometer.			

Figure 1: Comparing surface and logical-level outputs.

This survey therefore treats LT2T as a unique frame of reference for reasoning-centric generation research. Following background on LT2T and reasoning (Section 2), we review the primary datasets developed for LT2T (Section 3). We outline the key challenges that arise in this task (Section 4), and organize existing methods into a unifying taxonomy (Section 5). Our taxonomy links challenges to methods, enabling clearer comparisons and identification of underexplored directions. We then compare these methods (Section 6), discuss how to select them (Section 7), present future directions (Section 8), and outline conclusions (Section 9). In addition, we provide appendices that cover models (Appendix A), datasets and evaluation metrics (Appendix B), common logic types and functions (Appendix C), and further method comparisons

Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 1663–1677

(Appendix D). We hope this survey serves both as a roadmap for LT2T and as a blueprint for reasoning-aware generation more generally.

2 Background

This section briefly defines the Logical Table-to-Text task and discusses the types of reasoning used.

2.1 Logical Table-to-Text

The *Logical Table-to-Text* (LT2T) task is to learn a mapping from a table to an articulate natural language sentence that can be derived from the input table. Formally, the task can be defined as follows (Chen et al., 2020a):

Given a table T denoted as $T = \{ T_{i,j} \mid 1 \leq i \leq R_T,\ 1 \leq j \leq C_T \}$, where R_T and C_T are the number of rows and columns, respectively, and each cell entry $T_{i,j}$ may contain a word, a number, a phrase, or even an entire sentence; and reference sentence(s) of the form $W = (w_1, w_2, \ldots, w_n)$ composed of words w_i; the objective is to train a model $P(W \mid T)$ that generates a hypothesis $\hat{W}$ that is both (i) *fluent* and (ii) *logically entailed* by the information in T.

2.2 Reasoning

Reasoning involves analyzing facts to infer new insights, draw conclusions, or make decisions by identifying patterns, comparisons, or causal relationships within data. In LT2T, reasoning enables the generation of logically correct statements that extend beyond surface-level information. This survey focuses specifically on two types of reasoning:
Logical: *Logical reasoning* is inferring analytical relationships such as comparisons, superlatives, or causal connections. For example, given statements "A is taller than B" and "B is taller than C," logical reasoning can conclude that "A is taller than C."
Numerical: *Numerical reasoning* refers to making correct inferences from numerical data, including arithmetic operations, magnitude comparison, and numeric aggregation. For example, determining the tallest individual is A given specific heights {A: 180 cm, B: 170 cm, C: 160 cm}.

Although numerical reasoning is a subset of logical reasoning, it is considered separately here due to the specialized challenges it presents for LT2T.

3 Datasets

Training a generative model for LT2T requires a specific dataset, which typically consists of pairs of tables and narratives that describe table contents. Numerous datasets exist for generating descriptive text from tables, such as WikiBio (Lebret et al., 2016). However, our focus is on datasets specifically designed to challenge and evaluate the logical/numerical reasoning capabilities of LT2T techniques. Table 1 summarizes LT2T datasets.

In the following, we first discuss what is currently considered in the design of existing LT2T datasets and then what is missing.

LogicNLG (Chen et al., 2020a) increases logical difficulty through open-domain, unconstrained schemas and diverse logical operations such as superlatives and comparisons. The dataset's tables span multiple domains, though the distribution is heavily skewed toward sports: approximately 35% of tables concern teams/players and 25% concern competitions, followed by entertainment and politics at roughly 15% each, with celebrity and science domains appearing far less frequently.

Logic2Text (Chen et al., 2020b) pairs tables with logical forms (LFs); it also *quantifies structural reasoning complexity* using LF graph size, including number of nodes and function nodes. Each logical form contains on average nine nodes, including about three function nodes. The dataset includes seven logic types: Count (2.0k) and majority (1.8k) are the most common, followed by unique (1.6k), superlative and aggregation (1.4k each). Ordinal and comparative (1.2-3k each) are less frequent. This distribution shows that Logic2Text emphasizes summarization and highlighting prominent rows, while explicit comparison and ranking operations are present but occur less often.

NumericNLG (Suadaa et al., 2021) and *SciGen* (Moosavi et al., 2021) target numerical reasoning by emphasizing arithmetic operations; *SciGen* further introduces three splits, with the hardest split containing longer, more analytical statements.

ContLOG (Liu et al., 2022) is a controllable dataset built on Logic2Text that replaces LFs with highlighted evidence cells to guide logical content selection; it also provides a pre-training subset.

LoTNLG (Zhao et al., 2023d) is an evaluation-only benchmark designed for zero-shot/prompted LLM testing, conditioning generation on nine reasoning types to probe type-specific difficulty.

All of the above operate on flat tables. *HiTab* (Cheng et al., 2022) makes the task harder by introducing hierarchical (multi-level header) tables and shows that schema hierarchy stresses alignment and reasoning.

Dataset	Tables	Cell	Num.	Pairs	Vocab	Text	Domain	Source	Reasoning	Schema	Cont.	Struct.
LogicNLG	7.3K	91	35	37.0K	122K	14	Open (Wiki)	Annot.	Rich (Logical)	Unlimited	No	Flat
Logic2Text	5.5k	64*	22*	10.7k	14k	16.7	Open (Wiki)	Annot.	Rich (Logical)	Unlimited	No	Flat
NumericNLG	1.3K	35*	31*	1.3K	19.6K	94	Scientific	Hybrid	Rich (Numerical)	Unlimited	No	Flat
SciGen	—	53	34	1.3K	11K	116	Scientific	Annot.	Rich (Numerical)	Unlimited	No	Flat
ContLOG	5.5K	64*	22*	10.7K	14K*	16.7*	Open	Annot.	Rich (Logical)	Unlimited	Yes	Flat
LoTNLG	862	84*	25*	4.3K	6.3K*	14*	Open	Annot.	Rich (Logical)	Unlimited	Yes	Flat
HiTab	3597	190*	116*	10.6K	8.7K*	17.32*	Open	Annot.	Rich (Numerical)	Unlimited	Yes	Hier.

Table 1: Logical Table-to-Text Datasets.Tables = number of tables; Cell = avg total cells/table; Num = avg numeric cells/table; Pairs = number of annotated pairs; Vocab/ Text = vocabulary size / avg description length (words). Source = data origin (web, human-annotated, or hybrid). Schema = known (fixed columns/order) vs. unlimited (arbitrary columns). Cont. = guidance on what to verbalize e.g. highlighted cells. Struct. = table structure (flat or hierarchical). * = computed from released data.

Gaps and Directions. Despite these contributions, current LT2T datasets leave room for growth in at least three ways: (1) *Schema realism:* include hierarchical headers and *multi-table joins* to reflect real reports and dashboards. (2) *Operation coverage:* cover more operations such as temporal (trends, deltas, rolling statistics). (3) *Complexity metadata:* release per-dataset distributions such as operation types, domains, and numeric density.

4 Challenges in LT2T

This section discusses key challenges that have been addressed by existing research, particularly to improve LT2T.

Logical Fidelity: A generated sentence has *perfect fidelity* when every conclusion it makes is entailed by the information in the table. This means that each claim must logically and necessarily follow from the premises provided in the table. If a conclusion cannot be derived solely from the information presented, the text lacks fidelity. Common causes of low fidelity include: (1) missing or incorrect logical operations, (2) a mismatch between the sequence and logical order, and (3) over-reliance on superficial correlations rather than causal relationships grounded in the table.

Logical Controllability: The space of possible valid descriptions is exponentially large, as multiple logical inferences can be made from the same table. Current models struggle to determine which logical operation to apply, leading to irrelevant or uncontrolled outputs. Without guidance, models may select incorrect logic or reasoning paths. Therefore, it matters how models decide what content to include when multiple valid statements can be derived from a table.

Numerical Reasoning: Models commonly used in T2T, such as Pre-trained Language Models (PLMs) or general-purpose Large Language Models (LLMs), are trained as text token predictors rather than explicit numerical reasoners. As a result, they often exhibit weaknesses in tasks that require one or more of these numeric competencies: (1) *Magnitude Understanding*: Magnitude indicates the size of a number and is used to compare or order values; (2) *Arithmetic/Functional Operations*: Performing exact or approximate operations such as addition, subtraction, ratios, and aggregates; (3) *Number to Word Mapping*: Choosing appropriate lexical descriptors.

Data Scarcity: The success of many LT2T methods depends on the availability of large amounts of annotated data. However, annotation is an expensive task. Therefore, developing methods to enable text generation with few-shot samples and reducing annotation costs are critical.

Diversity: Models often focus on the same table regions or apply the same logical operations, leading to repetitive outputs. To promote diversity, it is essential to generate multiple distinct yet factually accurate statements derived from a table.

User Preference: Different users may want different logical views, such as trends vs. outliers. Ignoring this leads to mismatched summaries. Conditioning generation on user intent improves relevance and reasoning selection.

Evaluation Methodology: Conventional automatic metrics such as BLEU (Papineni et al., 2002) reward surface-level token overlap, so they miss logical inconsistencies and hallucinations. They also lack explanations as to why outputs are (in)correct. Consequently, the field needs logic-aware, explainable metrics that can both assess and justify a model's reasoning accuracy.

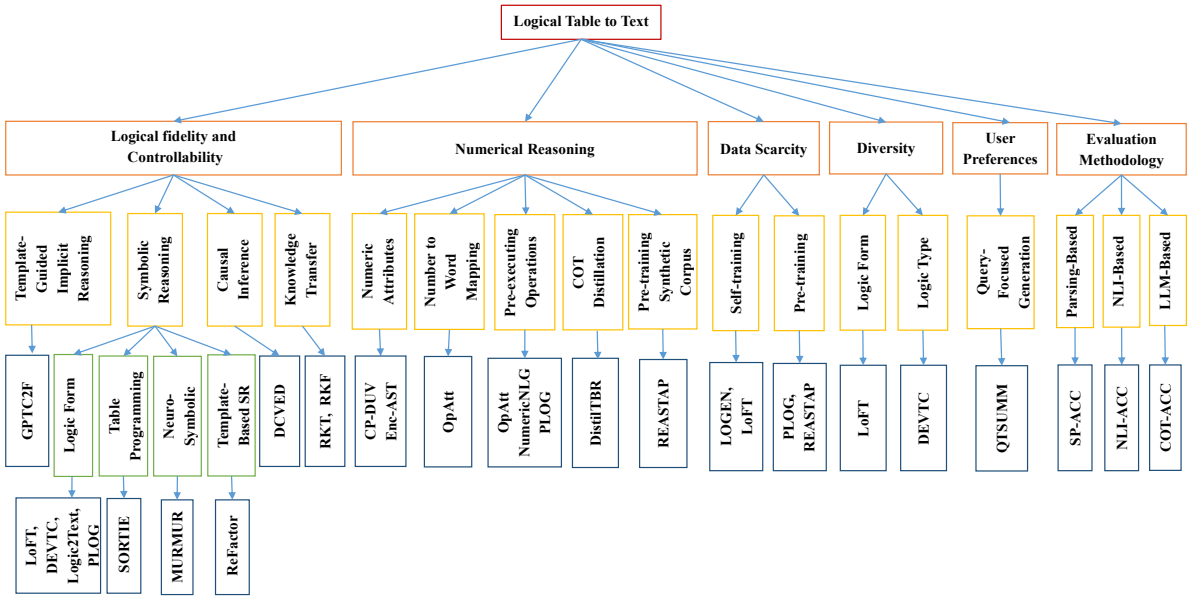

Figure 2: Taxonomy of logical table-to-text methods

5 Methods for Logical Table-to-Text

This section groups methods by the challenges they address; each method might appear in multiple categories. Figure 2 visualizes this: orange boxes mark key issues from Section 4, yellow and green boxes show main method families, and blue boxes list representative approaches. The diagram also outlines the order of the subsections that follow.

5.1 Logical Fidelity and Controllability

This section reviews papers that propose approaches to enhance logical fidelity and faithfulness in LT2T. Because many of these methods simultaneously enhance controllability, we discuss approaches for both issues here.

5.1.1 Template-Guided Implicit Reasoning

Auto-regressive generators emit tokens strictly left-to-right; however, logical reasoning can require performing step C before step B. This misalignment between language order and logical order can introduce reasoning errors (Chen et al., 2020a). To address this issue, Chen et al. (2020a) propose a two-stage coarse-to-fine decoder known as GPT-C2F. First, the model generates a template by replacing entities and numbers with placeholders to plan the logical structure. After fixing the global structure, it generates the final sentence by conditioning on both the table and its template to select the correct entity or number for each placeholder. Because the second pass treats the entire template as fixed context, each placeholder attends to its left and right neighbors and the connective words. This allows the model to infer the intended operation before selecting a value, thereby reducing order-mismatch errors.

5.1.2 Symbolic Reasoning-Based Methods

Symbolic reasoning (SR) introduces an explicit intermediate representation, such as logic forms (LFs), programs, or modular operators that specify exact operations to perform before text generation. By dividing a reasoning task into a sequence of clear, executable subtasks, SR approaches can validate each step in isolation and discard logically unsound paths before generating text. Because the final output is grounded in a verified chain of operations, these methods substantially improve logical fidelity and overall faithfulness.

Logic-Form–Based Methods: This subsection reviews methods that utilize logic-forms during different phases of their approach, such as pre-training, training, or inference to address fidelity.

Logic2Text (Chen et al., 2020b) and LoFT (Zhao et al., 2023b) both condition text generation on a table and its logic form. In both methods, the LF guides which table region and operation to reference, reducing ambiguity, and improving control over content selection. Logic2Text uses gold, hand-annotated Python-style LFs whose execution has

already been validated on the table; no extra checking is required. However, LoFT builds LFs automatically via a Structure-Aware Semantic Parsing (SASP) approach (Ou and Liu, 2022) at training time and a synthesis pipeline (Liu et al., 2022) at inference. Because automatically generated LFs can be noisy, LoFT also treats them as fact verifiers: each candidate LF must execute to True, and a Natural Language Inference (NLI)-based method filters out any sentence that the table does not entail.

PLOG (Liu et al., 2022) utilizes LFs as pretraining signals to teach logical reasoning. The idea is that by first training LFs, the model develops a deeper understanding of logical structures, which results in fewer logical errors in final text generation. In pretraining, a large synthetic dataset of (table, LF) pairs is used. The model is then fine-tuned to generate statements from tables. This approach improves controllability by explicitly highlighting which table cells are involved in each LF.

Table-Compatible Programming Language: SORTIE (Zhao et al., 2023a) improves logical reasoning by decoupling language generation from reasoning. It builds on Chen et al. (2020a)'s two-stage approach of generating a template, but fills each template placeholder via symbolic program execution instead of neural prediction. For every placeholder, a Gated Recurrent Unit (GRU) (Chung et al., 2014) scheduler picks an order based on logical dependencies, and another GRU generates a short, table-compatible program composed of operators and operands whose execution on the table returns the exact value and prevents hallucination. To handle missing annotations, SORTIE uses heuristic pseudo-labels with self-adaptive training. This clear division between *what to say* via template and *how to reason* by programs enhances logical fidelity and faithfulness.

Neuro-Symbolic Modular Reasoning: MURMUR (Saha et al., 2023) is a neuro-symbolic, modular framework designed to improve logical reasoning by explicitly separating symbolic logical reasoning from linguistic generation. Its core idea is to dynamically construct executable reasoning paths composed of symbolic and neural modules, governed by grammar and guided by a saliency-based value function during a best-first search. By explicitly performing logical operations using symbolic modules and restricting permissible compositions via grammar rules, MURMUR ensures each reasoning step is valid and verifiable. This approach directly addresses pitfalls such as hallucination, order mismatches, and semantic inconsistencies. Once a valid reasoning path is constructed, a language model converts it into a natural sentence. This method improves both the accuracy and faithfulness of generated summaries by making the reasoning process explicit and reliable.

Template-Based Symbolic Reasoning: ReFactor (Zhao et al., 2023c) explicitly retrieves and generates fact-guided reasoning signals. A set of predefined templates is designed to target reasoning skills such as numerical comparisons, aggregations, and conjunctions. These templates are instantiated and executed over tables using a fact generator to produce multiple facts. Relevant facts are ranked based on user input and included as signals to the model. This improves factual accuracy by providing explicit symbolic reasoning during generation.

5.1.3 Causal Inference

DCVED (Chen et al., 2021) addresses logical inconsistency caused by *hidden confounders*, unobserved factors that create spurious correlations between the input table and the output text (Keith et al., 2020). To resolve this, DCVED applies causal intervention using do-calculus (Pearl, 2010), shifting the learning objective from $p(y \mid x)$ to $p(y \mid \text{do}(x))$, thereby reducing the influence of confounders. To implement this, DCVED frames the generation process using a causal graph with two key variables: (1) Mediator z_m: information extracted from the table, (2) Confounder z_c: a variational latent representing misleading patterns such as frequent but irrelevant table entities. These latent spaces are supervised to be meaningful: z_m is guided by entities mentioned in the target sentence, while z_c is trained to predict unused but high-frequency distractors. During inference, DCVED samples multiple sentences by varying z_c and uses a trained model to select the most factually consistent output. By combining causal reasoning with variational modeling, spurious correlations are reduced, and logical fidelity is improved.

5.1.4 Knowledge Transfer

Liu et al. (2024) employs Reasoning Knowledge Transfer (RKT) to improve logical fidelity in LT2T. They fine-tune a LLaMA-2-6B as a transfer model on Logic2Text to produce natural-language logical rules from tables, synthesize such rules for LogicNLG, and then train a two-stage BART system: a reasoning module that predicts rules from

tables and a summary module that generates text conditioned on those rules—making outputs table-entailed rather than correlation-driven. Noting that some transferred rules are noisy, Bai et al. (2025) introduces the Reasoning Knowledge Filter (RKF) that employs a clean-up stage. RKF uses GPT-4o to annotate a subset of the training data for reasoning correctness. This subset is used to train a BART-large classifier that filters low-quality reasoning traces from the rest of the dataset. This filtering process improves SP-Acc (+1.4) and NLI-Acc (+0.7) by ensuring the generator only sees high-quality, table-consistent reasoning paths.

5.2 Numerical Reasoning

This section reviews methods that address numerical reasoning issues discussed in Section 4.

5.2.1 Understanding Numeric Attributes

CP-DUV (Gong et al., 2020) targets neural models' poor grasp of numerical magnitude caused by treating numbers as word tokens through two key upgrades: (1) They inject magnitude-aware embeddings. A transformer is pre-trained to rank every pair of numbers within a column; each number token is replaced at runtime by an embedding that knows both its size and realistic context. (2) They add a content-aware verification reward—a policy-gradient signal that rewards summaries containing correct entities and statistics in logical order while penalizing omissions and redundancies. These additions give the model a true *sense of numbers*, reducing magnitude errors and ordering output.

Enc-AST (Li et al., 2021) enhances numerical reasoning by addressing magnitude, relative importance, and inter-entity relationships. A hierarchical encoder incorporates two auxiliary tasks: (1) *Number ranking* captures column-wise magnitude; (2) *Importance ranking* models row-wise importance of numerical values. These tasks are learned through self-attention and fused via a gating mechanism. Additionally, a graph-based reasoning module models relationships between entities and enables reasoning over inter-entity relationships. These components enhance the model's ability to generate factually accurate and numerically grounded summaries.

5.2.2 Number to Word Mapping

OpAtt (Nie et al., 2018) addresses the issue of mapping numbers to words using a quantization layer that groups scalar values into a small number of learnable bins. A linear layer maps numbers into logits, then a softmax layer converts the logits into a probability distribution over bins, each with its own embedding. The final embedding is a weighted sum of all embeddings. This method enables the model to generalize across similar values and produce magnitude-aware words during text generation.

5.2.3 Pre-executing Operations

These methods pre-execute numerical operations and feed the results to the model to enhance its numerical reasoning. This reduces the burden of calculation and turns numeric operations that neural LMs struggle with into plain sequence tokens or features that models can copy or attend to.

OpAtt (Nie et al., 2018) pre-executes operations such as *minus* for score gaps and *argmax* for identifying the top scorer. The operations and results are encoded beside records and fed into the decoder. The decoder uses dual attention over operations and records, with a gating mechanism to decide when to prioritize operations or records. Additionally, the copy mechanism ensures outputs can originate from table cells, improving faithfulness.

Suadaa et al. (2021) pre-computes *max*, *min*, and *diff* for the target rows, stores them in a dedicated operation table (T_{OP}), and trains a copy-augmented model that uses placeholders, such as <*header max*>. At inference, a ranking-and-memory procedure replaces each placeholder with a value drawn from T_{OP} or the original data table, ensuring every numeric value in the output is faithfully copied rather than hallucinated.

PLOG (Liu et al., 2022) computes column-wise *sum*, *avg*, and *per-cell rank* values and inserts them into the flattened table structure, allowing the model to access them naturally during generation.

These methods demonstrate that pre-execution, alongside attention, gating, or copying, substantially improve models' numerical reasoning.

5.2.4 Chain-of-Thought Distillation

DistilTBR (Yang et al., 2024) transfers numerical-reasoning skills from an LLM to smaller PLMs by first having the teacher model produce step-by-step reasoning traces for each table. These traces, appended to the table, serve as supervised signals when fine-tuning the student model, guiding it to look at the right rows/columns and apply the correct arithmetic operations. This distillation yields a compact generator that mimics the teacher's logical reasoning without altering the

underlying architecture.

5.2.5 Pre-training with Synthetic Corpus

REASTAP (Zhao et al., 2022) injects diverse reasoning skills into a seq2seq model via synthetic QA pairs during pre-training, without relying on table-specific architectures. Conditioned on a serialized table and question, the model learns to align columns, filter rows, and execute numerical operations purely through its parameters.

5.3 Data Scarcity

Several recent methods aim to reduce the reliance on large amounts of annotated data by generating supervision from unlabeled or synthetic sources.

One approach is to create synthetic corpora offline, allowing models to pre-train before being fine-tuned on limited gold data. For example, PLOG uses handcrafted logic-form templates filled with table values. Instances are kept only if the generated logic-form evaluates to True, producing a large, automatically verified dataset. Similarly, REASTAP (Zhao et al., 2022) creates synthetic question-answer pairs using a template-based example generator (Yoran et al., 2022) for various reasoning skills over tables. This synthetic generation allows the model to learn effectively without relying heavily on human-annotated data.

A second line of work focuses on self-training via pseudo-labeling, where models expand a small seed set using unlabeled data. LOGEN (Deng et al., 2023) begins with a tiny annotated set and trains two GPT-2 models: one to map tables and logic forms to text; one in reverse. The reverse model generates logic forms for new (table, text) pairs, and mutual checks retain consistent examples that are used to retrain the models in a bootstrap loop.

Finally, on-the-fly weak supervision eliminates the need for any gold logic forms. LoFT synthesizes multiple logic-form candidates per table at training and inference time. An NLI-based verifier filters out any that aren't supported by the table, allowing the model to learn from verified candidates without manual annotation.

5.4 Diversity

LoFT and DEVTC (Perlitz et al., 2022) both improve diversity by enumerating alternate reasoning structures. At inference, LoFT creates multiple candidate logic forms per table from 45 templates across eight operation categories, while DEVTC samples from various logic types. Because each logic form or type focuses on a different operation or table region, the models generate a broad set of distinct, valid statements.

5.5 User Preference

QTSUMM (Zhao et al., 2023c) extends LT2T by tailoring generated summaries to user preferences. To achieve this, they fine-tune models on their *QTSUMM* dataset of query–summary pairs. During training, the model is conditioned on user queries alongside the table, guiding generation and producing summaries that directly address user-specific information needs. Additionally, ReFactor (Section 5.1.2) generates query-relevant facts, which are concatenated to the input of the models or used as prompts in LLM during fine-tuning and inference. These facts provide explicit reasoning evidence, further guiding the model to address user queries in the summary with high analytical fidelity.

5.6 Evaluation Methodology

This section reviews automatic metrics for evaluating logical fidelity and explainability.

Parsing-Based Evaluation: This metric evaluates logical fidelity by converting each generated sentence into an executable logic form and running it against the source table. The generated text is converted to the candidate logic forms using a weakly supervised semantic parser (Liang et al., 2009) via breadth-first search. The candidates are ranked based on consistency with the original sentence. The top-ranked logic form is selected and executed on the table. The generated statement is considered logically faithful if the result is `True`. *Semantic Parsing Accuracy* (SP-Acc) (Chen et al., 2020a) is the proportion of sentences whose top logical form evaluates to `True`. Its reliability depends on the parser quality; misparsing or language ambiguity can lead to false outcomes.

Natural Language Inference (NLI)-Based Evaluation: This metric verifies whether a given statement is true, false, or neutral based on the table. The key point is to utilize an NLI model trained to predict the logical relationship to measure the entailment score between the table and the generated text. The ratio of entailed is computed and used to approximate the model's fidelity. NLI-Acc (Chen et al., 2020a), TAPEX-Acc, and TAPAS-Acc (Liu et al., 2022) are examples of this evaluation metric.

Reference-Free, LLM-based Evaluation: Chain-of-Thought (CoT)-Acc (Zhao et al., 2023d) uses a 2-shot chain-of-thought prompt with GPT-3.5/4 to

Method	Backbone	Family	SP-Acc ↑	NLI-Acc ↑
Field-Infusing (Chen et al., 2020a)	Transformer	-	38.9	57.3
GPT-TabGen (med) (Chen et al., 2020a)	GPT-2	-	45.5	73.3
GPT2-C2F (med) (Chen et al., 2020a)	GPT-2	Template-Guided	45.3	76.4
DCVED (Chen et al., 2021)	GPT-2	Causal Inference	43.9	76.9
DEVTC (Perlitz et al., 2022)	GPT-2	Logic Form	45.6	77.0
PLOG (Liu et al., 2022)	BART-Large	Logic Form	50.5	88.9
LoFT (Zhao et al., 2023b)	BART-Large	Logic Form	57.7	86.9
REASTAP (Zhao et al., 2022)	BART-Large	Pretraining Synthetic	54.8	89.2
SORTIE (Zhao et al., 2023a)	BART-Large	Table Programming	57.8	89.3
RKT (Liu et al., 2024)	BART-Large	Knowledge Transfer	59.6	88.1
RKF (Bai et al., 2025)	BART-Large	Knowledge Transfer	61.0	88.8

Table 2: Comparison of LT2T Models on Logical Fidelity Metrics. Shows the best results reported in the original papers.

label each generated sentence as entailed or refuted with respect to the table, then reports accuracy. It yields the best human correlation among automated faithfulness metrics on LogicNLG.

6 Comparing Methods

We compare representative methods across three key dimensions: performance, efficiency, and interpretability/controllability. Since our focus is on logical reasoning, we evaluate performance primarily using logical fidelity metrics in Table 2.

Pre-trained vs Traditional Backbones: Overall, pre-trained backbone models consistently outperform traditional encoder–decoder architectures. For example, even TabGen, which is only fine-tuned on LogicNLG, surpasses field-infused Transformer models in logical fidelity metrics.

Template-Guided Implicit Reasoning: GPT-C2F, which utilizes coarse-to-fine generation, yields higher NLI-Acc and Adv-Acc than GPT-TabGen at both small and medium scales.

Logic-Form/Logic-Type Control: Human evaluations (Chen et al., 2020b) reveal a dramatic gain in factual correctness when logical forms are used, jumping from 20% without LFs to over 82% with LFs. DEVTC improves on prior GPT-2 baselines by conditioning generation on predicted logic types. While it gains slightly in logical metrics, its main strength lies in diversity: by switching logic types during generation, it achieves a factuality–diversity trade-off that surpasses the GPT-TabGen sampling frontier without requiring stochastic decoding, using greedy decoding alone. LoFT builds on this by conditioning on full executable logic forms and applying a verifier to filter outputs. This enables LoFT to achieve the best overall balance between faithfulness and diversity in this family. The trade-off, however, is increased implementation complexity: DEVTC requires a logic-type classifier, while LoFT involves a full pipeline with logic-form parsing, synthesis, and verification. These methods offer improved controllability and more transparent reasoning processes.

Implicit Skill Injection (Pretraining): PLOG and REASTAP inject logical reasoning through pretraining; PLOG leverages logical forms (LFs), while REASTAP uses 4 million synthetic question–answer pairs. Both significantly improve logical fidelity metrics. These approaches keep inference simple at the expense of pretraining cost; for instance, REASTAP's pretraining required 34 hours on an 8 NVIDIA A5000 24GB cluster. They improve faithfulness through pretraining signals, but the final generation remains purely neural, so the internal reasoning steps aren't explicitly exposed.

Table Programming: SORTIE reports the strongest NLI-Acc among compared methods. It outperforms PLOG and REASTAP without any large-scale reasoning pretraining—training for roughly 5 hours on 8×3090 Ti GPUs—while offering high interpretability via explicit, executable reasoning traces. The trade-off is a higher implementation burden.

Knowledge Transfer: RKT and RKF report the strongest logical-fidelity metrics among peers. Both are highly interpretable and controllable via explicit rules.

Not all models are evaluated on these metrics; we discuss other models that use other logical metrics in Appendix D.

7 Discussion

This survey found two dominant LT2T paradigms: (1) **Explicit trace construction:** logic-forms, programs, or neuro-symbolic plans that decouple reasoning from surface realization and give strong fidelity guarantees. (2) **Implicit skill injection:** pretraining or distillation schemes that embed numerical or logical competence inside neural parameters with minimal task-specific engineering.

Explicit methods appear better when one needs: (a) auditability/controllability; (b) verifiable, stepwise reasoning; or (c) operator-level control. However, we suggest *implicit methods* when (a) simple, fast inference at scale is required; (b) there is lack of annotations or tooling for logic forms; or (c) one needs broad skill coverage from synthetic data.

8 Future Directions

We suggest four primary avenues for advancing LT2T research, considering current limitations and existing challenges.

Improving Reasoning Capabilities: Existing approaches are limited to a few operations, such as *sum*, *min*, or *max* (Appendix C), and struggle with more advanced analytical needs, such as trend detection or multi-hop comparisons. Research should address this limitation through three main thrusts: (1) Support more complex logical skills for advanced analytics,including change rates, graph traversal, and temporal patterns. (2) Develop flexible logical supervision frameworks that help models acquire these skills without explicit structured logical forms. Lightweight, weakly supervised, or self-training methods may allow models to infer logical structures from training data without requiring gold-standard logic annotations. (3) Include dynamic logical inference, where the model constructs and executes operator chains during inference to ensure faithful intermediate results before verbalization. Approaches may range from symbolic search with learned value functions to prompting LLMs to produce executable code. Together, these directions will enable broader coverage of analytical reasoning skills, stronger factual accuracy, and adaptability to evolving data schemas.

Evaluation Metrics: We need metrics that are both logic-aware and explainable. While metrics like COT-Acc offer step-by-step reasoning to judge entailment, they are costly, non-deterministic, and limited to proprietary APIs. We need open, lightweight, logic-aware metrics that break outputs into table-grounded claims and mark each as *entailed*, *contradicted*, *missing*, or *hallucinated*.

Scalability: Future LT2T systems must adapt to evolving computational resources and table schemas. New schemas often demand novel logical skills, such as rate-of-change analysis or multi-row trend detection, which current models struggle to acquire without retraining. Exploring continual or incremental learning frameworks that allow models to acquire new reasoning skills while retaining prior knowledge appears promising. Moreover, the high training and inference costs of LLMs, together with their fixed context-window limits, make them impractical for many real-world deployments. Research should thus prioritize lightweight, modular, or distilled models that retain strong logical-reasoning capabilities while reducing computational and memory requirements.

Applicability: Current models are designed for flat tables, whereas real-world data often involves heterogeneous formats like event sequences, hierarchical tables with multi-level headers, or nested structures. Future LT2T systems should be able to process complex structured data and reason over temporal, hierarchical, or relational formats, as seen in domains like medicine or finance. There is also a need for multi- and cross-lingual LT2T generation, where models maintain logical reasoning consistency across languages. Logical operations such as arithmetic, comparison, causal inference, and multi-hop deduction must transfer cleanly across linguistic boundaries. Expanding LT2T to low-resource languages, global reporting, and multilingual scientific domains will promote greater robustness, generalization, and cross-domain utility.

9 Conclusions

Logical Table-to-Text (LT2T) has become a microcosm of reasoning-aware generation-structured inputs, explicit operations, and verifiable outputs. Explicit methods boost fidelity and interpretability, while implicit ones scale better; hybrid models that pair symbolic checks with distilled reasoning show promise in balancing both.

Though focused on LT2T, these insights extend to broader text generation: reasoning traces, logic-aware supervision, and fidelity-based evaluation are key for summarization, retrieval-augmented, and multi-modal tasks. Future progress lies in richer operator coverage, explainable metrics, and lightweight continual-learning frameworks.

Limitations

This survey focuses on LT2T as a lens for examining reasoning in natural language generation. While this scope allows for a deep and actionable taxonomy of methods, it excludes other important forms of reasoning such as commonsense, causal, and spatial. Furthermore, we primarily review methods from the LT2T domain; techniques from related tasks such as fact verification, question answering, and code generation could offer additional insights and warrant future exploration.

Acknowledgments

We used an AI assistant (ChatGPT) for language editing (clarity and grammar). All content was verified and edited by the authors against the cited papers. No confidential or personal data were provided to the tool. The assistant did not generate novel experimental results or datasets.

References

Yu Bai, Baoqiang Liu, Shuang Xue, Fang Cai, Na Ye, and Guiping Zhang. 2025. Reasoning knowledge filter for logical table-to-text generation. In *Proceedings of Bridging Neurons and Symbols for Natural Language Processing and Knowledge Graphs Reasoning @ COLING 2025*, pages 18–30, Abu Dhabi, UAE. ELRA and ICCL.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901. NeurIPS 2020.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 7929–7942, Online. Association for Computational Linguistics.

Wenqing Chen, Jidong Tian, Yitian Li, Hao He, and Yaohui Jin. 2021. De-confounded variational encoder-decoder for logical table-to-text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL 2021), Long Papers*, pages 5532–5542, Online. Association for Computational Linguistics.

Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang. 2020b. Logic2Text: High-fidelity natural language generation from logical forms. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2096–2111, Online. Association for Computational Linguistics.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2022. Hitab: A hierarchical table dataset for question answering and natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Long Papers*, pages 1094–1110, Dublin, Ireland. Association for Computational Linguistics.

Junyoung Chung, Çağlar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*. ICLR 2015 Workshop Track.

Shumin Deng, Jiacheng Yang, Hongbin Ye, Chuanqi Tan, Mosha Chen, Songfang Huang, Fei Huang, Huajun Chen, and Ningyu Zhang. 2023. LOGEN: Few-shot logical knowledge-conditioned text generation with self-training. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2124–2133.

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*, pages 2905–2914, Online. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Katherine Keith, David Jensen, and Brendan O'Connor. 2020. Text and causal inference: A review of using text to remove confounding from causal estimates. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 5332–5344, Online. Association for Computational Linguistics.

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 (EMNLP 2016)*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke 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 (ACL 2020)*, pages 7871–7880, Online. Association for Computational Linguistics.

Liang Li, Can Ma, Yinliang Yue, and Dayong Hu. 2021. Improving encoder by auxiliary supervision tasks for table-to-text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL 2021), Long Papers*, pages 5979–5989, Online. Association for Computational Linguistics.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL 2009)*, pages 91–99, Suntec, Singapore. Association for Computational Linguistics.

Ao Liu, Haoyu Dong, Naoaki Okazaki, Shi Han, and Dongmei Zhang. 2022. PLOG: Table-to-logic pretraining for logical table-to-text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022)*, pages 5531–5546, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Baoqiang Liu, Yu Bai, Fang Cai, Shuang Xue, Na Ye, and Xinyuan Ye. 2024. Reasoning knowledge transfer for logical table-to-text generation. In *Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Yokohama, Japan. IEEE.

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 (AAAI-18)*, pages 4881–4888. AAAI Press.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. SciGen: A dataset for reasoning-aware text generation from scientific tables. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Track on Datasets and Benchmarks*. Datasets and Benchmarks Track (Round 2).

Feng Nie, Jinpeng Wang, Jin-Ge Yao, Rong Pan, and Chin-Yew Lin. 2018. Operation-guided neural networks for high fidelity data-to-text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 3879–3889, Brussels, Belgium. Association for Computational Linguistics.

Suixin Ou and Yongmei Liu. 2022. Learning to generate programs for table fact verification via structure-aware semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Long Papers*, pages 7624–7638, Dublin, Ireland. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Judea Pearl. 2010. On measurement bias in causal inference. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, pages 425–432, Catalina Island, California, USA. AUAI Press.

Yotam Perlitz, Liat Ein-Dor, Dafna Sheinwald, Noam Slonim, and Michal Shmueli-Scheuer. 2022. Diversity enhanced table-to-text generation via type control. *CoRR*, abs/2205.10938.

Raheel Qader, Khoder Jneid, François Portet, and Cyril Labbé. 2018. Generation of company descriptions using concept-to-text and text-to-text deep models: dataset collection and systems evaluation. In *Proceedings of the 2018 Workshop (WS2018 #65)*, pages 254–263, Tilburg University, The Netherlands. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Technical Report*.

Swarnadeep Saha, Xinyan Yu, Mohit Bansal, Ramakanth Pasunuru, and Asli Celikyilmaz. 2023. MURMUR: Modular multi-step reasoning for semi-structured data-to-text generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11069–11090, Toronto, Canada. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational*

Linguistics (ACL 2017), pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Lya Hulliyyatus Suadaa, Hidetaka Kamigaito, Kotaro Funakoshi, Manabu Okumura, and Hiroya Takamura. 2021. Towards table-to-text generation with numerical reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL 2021), Long Papers*, pages 1451–1465, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:6000–6010.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Bohao Yang, Chen Tang, Kun Zhao, Chenghao Xiao, and Chenghua Lin. 2024. Effective distillation of table-based reasoning ability from LLMs. In *Proceedings of the 2024 Conference on Language Resources and Evaluation (LREC 2024)*, pages 5538–5550, Torino, Italia. ELRA and ICCL.

Ori Yoran, Alon Talmor, and Jonathan Berant. 2022. Turning tables: Generating examples from semi-structured tables for endowing language models with reasoning skills. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Long Papers*, pages 6016–6031, Dublin, Ireland. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xueliang Zhao, Tingchen Fu, Lemao Liu, Lingpeng Kong, Shuming Shi, and Rui Yan. 2023a. SORTIE: Dependency-aware symbolic reasoning for logical data-to-text generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11247–11266, Toronto, Canada. Association for Computational Linguistics.

Yilun Zhao, Linyong Nan, Zhenting Qi, Rui Zhang, and Dragomir Radev. 2022. ReasTAP: Injecting table reasoning skills during pre-training via synthetic reasoning examples. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022)*, pages 9006–9018, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yilun Zhao, Zhenting Qi, Linyong Nan, Lorenzo Jaime Flores, and Dragomir Radev. 2023b. LoFT: Enhancing faithfulness and diversity for table-to-text generation via logic form control. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2023)*, pages 554–561, Dubrovnik, Croatia. Association for Computational Linguistics.

Yilun Zhao, Zhenting Qi, Linyong Nan, Boyu Mi, Yixin Liu, Weijin Zou, Simeng Han, Ruizhe Chen, Xiangru Tang, Yumo Xu, Dragomir Radev, and Arman Cohan. 2023c. QTSumm: Query-focused summarization over tabular data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*, pages 1157–1172, Singapore. Association for Computational Linguistics.

Yilun Zhao, Haowei Zhang, Shengyun Si, Linyong Nan, Xiangru Tang, and Arman Cohan. 2023d. Investigating table-to-text generation capabilities of large language models in real-world information seeking scenarios. In *Proceedings of the Industry Track at the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*, pages 160–175, Singapore. Association for Computational Linguistics.

A Models

We categorize the models commonly used in LT2T works into three groups.

Traditional Encoder-Decoder Models: Early works train encoder-decoders such as Gated Recurrent Neural Network (GRU) (Chung et al., 2014), Long Short Term memory (LSTM) (Hochreiter and Schmidhuber, 1997), or vanilla Transformers (Vaswani et al., 2017) from scratch. These models are usually augmented with structural add-ons, such as copy mechanisms (Gu et al., 2016) or pointer networks (See et al., 2017), gating (Liu et al., 2018), or attention mechanisms (Luong et al., 2015) to ensure numbers and entity names are reproduced faithfully. Methods that use these models are presented in Table 3.

Pre-trained Language Models (PLMs): Subsequent studies fine-tune generic PLMs, such as BART (Lewis et al., 2020), T5 (Qader et al., 2018), or GPT (Radford et al., 2018) on one or more LT2T datasets. Because these models have strong linguistic priors, only light task-specific tuning is required to outperform scratch baselines and traditional methods. Table 4 represents methods that utilize PLMs in their architecture.

Large Language Models (LLMs): A growing line of work leverages LLMs such as OPT (Zhang et al., 2022) and GPT-3/4 (Brown et al., 2020; Bubeck

Method	Base Model	Attention	Gating	Copy	Training Method
OpAtt (Nie et al., 2018)	GRU	Yes	Yes	Yes	Maximum Likelihood Estimation
CP-DU (Gong et al., 2020)	LSTM/Transformer	Yes	Yes	Yes	Reinforcement Learning
Field-Gating (Chen et al., 2020a)	LSTM/Transformer	Yes	Yes	Yes	Maximum Likelihood Estimation
Field-Infusing (Chen et al., 2020a)	LSTM/Transformer	Yes	No	Yes	Maximum Likelihood Estimation
DCVED (Chen et al., 2021)	Transformer	Yes	No	No	Causal Intervention

Table 3: Traditional encoder-decoders trained from scratch for LT2T.

Method	Base Model	Param. Size	Training Method
BERT-TabGen (Chen et al., 2020a)	BERT-*small*/*large*	140/340M	MLE
GPT-TabGen (Chen et al., 2020a)	GPT2-*small*/*medium*	117/345M	MLE; Adv-Reg; RL
GPT-C2F (Chen et al., 2020a)	GPT2-*small*/*medium*	117/345M	MLE
Fine-tuned GPT2 (Chen et al., 2020b)	GPT2-*small*	117M	FT
PLOG (Liu et al., 2022)	BART-*large*	406M	PT/FT
	T5-*base*/*large*	220/770M	PT/FT
REASTAP (Zhao et al., 2022)	BART-*large*	406M	PT/FT
DEVTC (Perlitz et al., 2022)	GPT2-*small*/*medium*	117/345M	Supervised
LoFT (Zhao et al., 2023b)	BART-*large*	406M	Supervised
LOGEN (Deng et al., 2023)	GPT2-*small*	117M	Few-shot self-training
DistilTBR (Yang et al., 2024)	Flan-T5-CoT-*base*/*large*	250/780M	FT
	T5-CoT-*base*/*large*	220/770M	
RKT (Liu et al., 2024)	BART-*large*	406M	FT
RKF (Bai et al., 2025)	BART-*large*	406M	FT

Table 4: Pre-trained language models fine-tuned for LT2T. MLE = Maximum Likelihood Estimation, RL = Reinforcement Learning, PT = Pre-training, FT = Fine-tuning.

et al., 2023) without task-specific fine-tuning. Instead of updating weights, researchers steer models via in-context learning, including zero- or few-shot exemplars (Brown et al., 2020), and Chain-of-Thought (CoT) prompting (Wei et al., 2022) to boost faithfulness.

Beyond prompting, LLMs act as teachers, synthesizing CoT rationales to train smaller students (DistilTBR) or to transfer reasoning knowledge (RKT), and as automated annotators whose labels supervise downstream classifiers (RKF). Table 5 summarizes these LLM-based strategies.

B Datasets and Evaluation Metrics

Table 6 shows the datasets and evaluation methodology used. Note that early studies evaluated their methods on datasets that are not specifically designed for LT2T, including RotoWire and MLB.

C Logic Types and Logic Functions

Most of the studies consider the same logic types and functions defined in (Chen et al., 2020b). The logic types include count, unique, comparative, superlative, ordinal, aggregation, and majority. For complete definitions and examples, please refer to the Appendix of that paper. The functions are presented in Table 7 of that paper.

D Comparison of Methods-Cont.

We compare methods in Section 6 primarily using SP-Acc and NLI-Acc, which are the most widely adopted logical fidelity metrics across prior work. However, not all studies report these measures. To ensure completeness, we summarize in Table 7 the best results of models evaluated with TAPAS-Acc and TAPEX-Acc, using the values reported in their original papers. We further provide a comparative rating of all surveyed methods along five axes—diversity, interpretability, controllability, implementation burden, and cost—in Table 8. The accompanying text details the rubric and justifies each assignment.

COT Distillation/LLMs: LLM-T2T (Zhao et al., 2023d) reports that GPT-family models outperform fine-tuned systems such as logic-form-based methods, including LoFT and PLOG. It also finds that adding more shots or Chain-of-Thought (CoT) prompting yields non-monotonic gains—more exemplars or CoT do not necessarily help; GPT-4-zero-shot performs better than GPT-4-1/2-COT. Zhao et al. (2023c) attain state-of-the-art TAPAS-Acc by augmenting LLMs with ReFactor. Finally,

Method	Base Model	Param. Size	Training Method
MURMUR (Saha et al., 2023)	OPT	175B	In-context learning
GPT-3.5 (Zhao et al., 2023d)	GPT-3.5	175B	In-context learning
GPT-4 (Zhao et al., 2023d)	GPT-4	-	In-context learning
RKT (Liu et al., 2024)	LLAMA2	6.7B	FT
DistilTBR (Yang et al., 2024)	GPT-3.5-turbo	175B	COT Distillation
RKF (Bai et al., 2025)	GPT-4o	-	Distillation

Table 5: LLMs used via prompting for LT2T.

Method	Dataset	Surface Metric	Logic Metric	Human Eval.
OpAtt	RotoWire	BLEU	RG	–
CP-DUV	RotoWire; MLB	BLEU	RG / CS / CO	–
Field-Gating	LogicNLG	PPL; BLEU-1,2,3	SP-Acc; NLI-Acc; Adv-Acc	Yes
Field-Infusing	LogicNLG	PPL; BLEU-1,2,3	SP-Acc; NLI-Acc; Adv-Acc	Yes
DCVED	LogicNLG; Logic2Text	BLEU	SP-Acc; NLI-Acc	–
BERT-TabGen	LogicNLG	PPL; BLEU-1,2,3	SP-Acc; NLI-Acc; Adv-Acc	No
GPT-TabGen	LogicNLG	PPL; BLEU-1,2,3	SP-Acc; NLI-Acc; Adv-Acc	Yes
GPT-C2F	LogicNLG	BLEU-1,2,3	SP-Acc; NLI-Acc; Adv-Acc	Yes
Fine-tuned GPT2	Logic2Text	BLEU-4; ROUGE-1,2,L	No	Yes
PLOG (BART/T5)	LogicNLG; CONTLOG	BLEU-1,2,3	SP-Acc; NLI-Acc; TAPEX/TAPAS-Acc	Yes
REASTAP	LogicNLG	BLEU-1,2,3	SP-Acc; NLI-Acc	No
DEVTC	LogicNLG	BLEU-1,2,3	SP-Acc; NLI-Acc	Yes
LoFT	LogicNLG	BLEU-1,2,3	SP-Acc; NLI-Acc	Yes
LOGEN	Logic2Text	BLEU; ROUGE-L	No	Yes
MURMUR	LogicNLG	BLEU; METEOR	No	Yes
DistilTBR	SciGen	METEOR; BERTScore; BLEURT	TAPAS/TAPEX-Acc	No

Table 6: Datasets and evaluation metrics used in surveyed papers.

DistilTBR demonstrates that distilling CoT-style supervision into smaller models like Flan-T5 and T5 can improve logical fidelity without relying on very large LLMs. MURMUR shows via human evaluation 26% more logically consistent summaries on LogicNLG vs direct prompting.

In the following, we explain how we rank each column in Table 8.

Diversity

Low: The model repeatedly generates similar outputs that rely on the same logical operation or remains confined to a fixed table region.

Medium: The model produces varied statements, but often focuses on similar regions or operations unless explicitly guided.

High: The model includes explicit mechanisms to diversify content selection, such as logic-type control, logic-form conditioning, or prompt variation, enabling it to describe different logic types and table regions.

Controllability

Low: There is little or no means to steer logical operations; the model explores a large, unconstrained search space.

Medium: Logic-type tokens, templates, or other signals allow partial steering of logic types but not specific operations or arguments.

High: The method allows explicit control via intermediate structures, including logic forms, programs, or verified candidates that deterministically guide generation.

Interpretability

Low: No explicit intermediate reasoning artifacts are available; the model operates as a black box.

Medium: Lightweight signals such as logic-type tags, CoT summaries, or templates

Method	Backbone	Family	TAPAS-Acc ↑	TAPEX-Acc ↑
PLOG (Liu et al., 2022)	T5-Large	Logic Form	76.0	75.9
LoFT (Zhao et al., 2023b)	BART-Large	Logic Form	-	61.8
LLM-T2T (Zhao et al., 2023d)	GPT-4-Zero-shot	LLM	91.8	91.0
	GPT-4-1-shot-direct		87.6	88.0
	GPT-4-1-shot-COT		89.4	90.8
	GPT-4-2-shot-direct		92.0	89.6
	GPT-4-2-shot-COT		88.8	90.4
ReFactor (Zhao et al., 2023c)	GPT4-zero-shot	Template-Based SR	92.3	-
	GPT3.5-1-shot		94.3	-
	GPT4-2-shot		93.3	-
DistilBTR (Yang et al., 2024)	FLan-T5-Base-COT	COT Distillation	78.72	82.75
	T5-Large-COT		80.62	81.97

Table 7: Comparison of LT2T Models on other logical fidelity Metrics.

Method	Diversity	Controllability	Interpretability	Implementation Burden
GPT-C2F	Med	Med	Med	Med
Fine-tuned GPT-2	Med	High	High	Med
DCVED	Low	Low	Low	Med
PLOG	Med	Med	Low	Med
REASTAP	Med	Med	Low	High
DEVTC	High	Med	Med	Med
LoFT	High	High	High	High
LOGEN	Med	Med	Med-High	Med
MURMUR	High	High	High	High
ReFactor	Med	Med	Med	Med
LLM-T2T	Med	Med	Med	Low
DistilBTR	Low-Med	Low	Med	Med
RKT	Med	High	High	Med-High
RKF	Med	High	Med-High	Med

Table 8: Comparing methods based on various criteria.

provide some interpretive cues but not complete reasoning traces.

High: The model produces executable or inspectable intermediates such as logic forms or symbolic programs that make the reasoning process transparent.

Implementation Burden

Low: Involves only prompt design or minimal fine-tuning.

Medium: Requires one auxiliary component (e.g., classifier, teacher model, or template mechanism) within a standard training pipeline.

High: Entails multi-stage pipelines (e.g., parser/synthesizer, generator, and verifier) or large-scale pretraining/data synthesis, often with custom executors.