

ConceptFormer: Towards Graph-Native Grounding of Large Language Models via Latent Concept Injection

Joel Barmettler
joel.barmettler@uzh.ch
University of Zurich
Zurich, Switzerland

Abraham Bernstein
bernstein@ifi.uzh.ch
Department of Informatics
University of Zurich
Zurich, Switzerland

Luca Rossetto
luca.rossetto@dcu.ie
School of Computing
Dublin City University
Dublin, Ireland

Abstract

Retrieval Augmented Generation (RAG) has enjoyed increased attention in the recent past, yet recent advancements in Large Language Models (LLMs) have highlighted the difficulty of ensuring trustworthiness and factuality. Current RAG methodologies often modify the internal architecture of pre-trained language models (PLMs) or rely on textifying knowledge graphs (KGs), which results in lossy linearization and context saturation. This paper introduces *ConceptFormer*, a neuro-symbolic approach to ground LLMs in structured knowledge from the Web of Data (e.g., Wikidata), without altering their internal structure or relying on textual input of KGs. *ConceptFormer* operates in the LLM embedding vector space, creating and injecting *concept vectors* that encapsulate the topological structure of the KG nodes directly. Trained in conjunction with a frozen LLM, *ConceptFormer* generates a comprehensive lookup table that maps KG nodes to their respective *concept vectors*. The approach aims to enhance the factual reliability of LLMs by enabling them to process these *concept vectors* natively, thus enriching them with structured world knowledge in a graph-native manner. Our experiments demonstrate that the addition of *concept vectors* to GPT-2 0.1B substantially increases its factual recall ability (Hit@10) by up to 272% when tested on sentences from Wikipedia and up to 348% on synthetically generated sentences. Even injecting only a single *concept vector* into the prompt increases factual recall ability (Hit@10) by up to 213% on Wikipedia sentences, significantly outperforming RAG with graph textification. This demonstrates that preserving topological structure in latent space is more effective for factuality than textual linearization, while coincidentally reducing token consumption by 130x.

CCS Concepts

• **Computing methodologies** → *Knowledge representation and reasoning*; • **Information systems** → *Data encoding and canonicalization*; *Language models*.

Keywords

Retrieval Augmented Generation, Knowledge-Graph Embedding, Knowledge Injection

ACM Reference Format:

Joel Barmettler, Abraham Bernstein, and Luca Rossetto. 2026. ConceptFormer: Towards Graph-Native Grounding of Large Language Models via Latent Concept Injection. In *Companion Proceedings of the ACM Web Conference 2026 (WWW Companion '26)*, April 13–17, 2026, Dubai, United Arab Emirates. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3774905.3794653>

1 Introduction

Large Language Models (LLMs) have demonstrated exceptional potential in various natural language processing tasks, including conversational agents, summarization, and information retrieval (IR). They are typically trained on large-scale, general-purpose (text) corpora and optimized via self-supervision objectives [21]. Through this pretraining, LLMs acquire substantial amounts of knowledge, which is implicitly stored within their model weights [38, 56]. However, such implicit storage can lead to a lack of factual grounding and risks of outdated, biased, or incomplete information, posing key obstacles for retrieval-augmented generation (RAG) and trustworthy AI systems [20, 25, 28].

A common approach to enhance knowledge retention in LLMs is through corpus curation or expansion, which has shown promise in reducing hallucinations [13, 46]. But simply improving or enlarging the training corpus does not always resolve deeper challenges such as domain-specific knowledge retrieval or real-time updates [14, 43]. Models still frequently hallucinate or omit key facts [29, 35], especially in specialized domains where domain shifts are inevitable [47]. Research has shown that growing the parameter count of LLMs can improve factual recall [18], but even large models like ChatGPT still struggle to faithfully reproduce structured knowledge, e.g., from DBPedia [40, 51].

To address these challenges in knowledge-intensive scenarios, Knowledge Graphs (KGs) [16, 30] have emerged as a valuable structured resource. KGs—such as Wikidata [45]—capture accurate, up-to-date, and domain-rich factual information. Making this graph-based knowledge accessible to LLMs is a longstanding goal, especially for retrieval-augmented generation and question answering [1, 53]. Popular RAG-focused approaches typically rely on “textification” of graph edges or nodes [10, 22], concatenating them into the LLM prompt. While effective, this results in a lossy linearization of the graph topology and consumes large portions of the model’s context window [55].

In this paper, we introduce *ConceptFormer*, a novel, neuro-symbolic way to integrate KGs into any pre-trained LLM without modifying its internal architecture or retraining its core parameters. Rather than textifying knowledge, *ConceptFormer* injects *concept vectors* derived from the KG directly into the LLM’s input embedding space. Figure 1 illustrates how entity recognition links a text mention



This work is licensed under a Creative Commons Attribution 4.0 International License. *WWW Companion '26, Dubai, United Arab Emirates*
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2308-7/2026/04
<https://doi.org/10.1145/3774905.3794653>

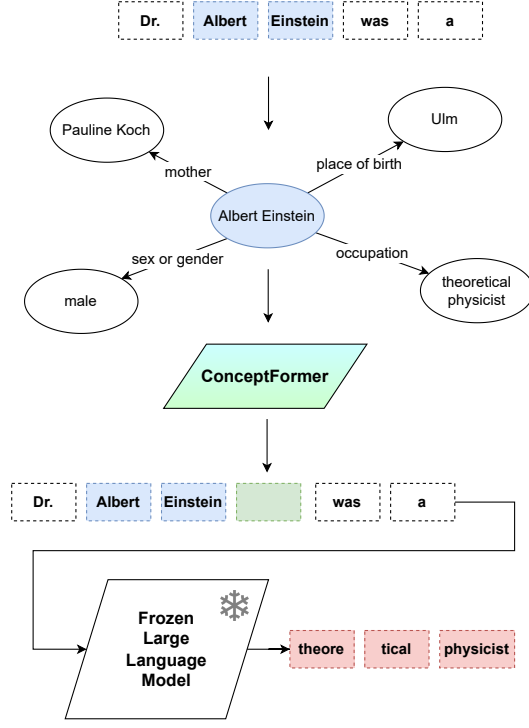


Figure 1: ConceptFormer enhances a prompt by extending the embedding vectors from the original prompt with learned concept vectors. It creates a vector embedding for the entity that is compatible with the LLMs input embedding space. The resulting concept vector (in green) captures the topological structure of the entity far better than the original token embedding vectors.

(“Albert Einstein”) to its corresponding Wikidata node, and *ConceptFormer* learns to produce a small set of dense *concept vectors* that augment or replace the naive token embeddings. These vectors *natively* encode the most relevant graph-neighborhood information, enabling the LLM to integrate structured knowledge with minimal context overhead.

The contributions of this paper are threefold: (1) We present *ConceptFormer*, a flexible mechanism to embed KG nodes into the LLM prompt space, enabling large-scale or domain-specific factual grounding without altering the LLM architecture. (2) We introduce new datasets—*Tri-REx*, *T-REx Bite*, and *T-REx Star*—specifically designed to evaluate next-token and factual-recall tasks. These datasets facilitate measuring how well LLMs retrieve, re-rank, and generate entity-level facts. (3) We empirically demonstrate that *ConceptFormer* achieves **up to 348% improvement** in factual recall (Hit@10) on synthetic sentences, and **up to 272% improvement** on Wikipedia-based sentences, compared to a GPT-2 0.1B baseline. Remarkably, even a *single concept vector* yields a competitive recall boost while consuming **130× fewer tokens** than text-based RAG. We make all datasets, our implementation of the model and dataset generators, as well as pre-trained models [2] freely available for download via <https://github.com/joelbarmettlerUZH/ConceptFormer>.

Whereas most KG-enhanced LLM research focuses on the *text representation of knowledge* [26, 41, 54], we focus on *vector-based injection* that compresses graph information directly within the LLM’s input space. We deliberately chose GPT-2 0.1B [37] for our experiments due to its comparatively small size and simple architecture. However, the technique extends to larger LMs with minimal adaptation.

2 Related Work

ConceptFormer intersects several research areas. We situate our work among (i) retrieval-augmented generation (RAG), (ii) KG-enhanced LLMs, (iii) latent representation learning, (iv) prompt tuning, and (v) pseudo-word embeddings for multimodal or specialized concepts.

Retrieval-Augmented Generation (RAG). Retrieval-augmented generation (RAG) [12, 20] enhances LLM-based text generation by retrieving relevant external data before or during text generation, thus grounding model outputs in factual information from structured or unstructured sources. In conventional IR pipelines, this approach is reminiscent of query expansion or contextualization: the user query (or partial text) is augmented with retrieved documents, enabling the LLM to generate content that more accurately reflects the retrieved evidence. A straightforward yet potent variant is *graph textification*, wherein a knowledge graph (KG) is linearized into textual templates [10]. For instance, every subject–predicate–object triple can be converted into a text phrase and concatenated with the user query. Works like Brate et al. [7] and Li et al. [22] rely on these textual expansions to harness structured knowledge in an LLM’s generation process. While effective, this strategy imposes heavy token overhead, straining context windows and obscuring the topological structure of the graph data. By contrast, *ConceptFormer* avoids this lossy linearization. Rather than inserting hundreds of tokens describing a KG neighborhood, it transforms the KG node (and its adjacency information) into a small set of dense, learned vectors directly injected into the LLM’s embedding space. This preserves the topological integrity of the evidence while minimizing context usage.

KG-Enhanced LLMs. Beyond RAG, multiple lines of research aim to integrate KG information into an LLM’s representation. Early works frequently required extensive architecture modifications or additional training heads. For example: *DKPLM* [54] dynamically updates language models with knowledge extraction and pseudo-token injections, altering model layers and unfreezing part of the network. *CoLAKE* [41] builds a hybrid word-knowledge graph and modifies both the embedding and encoder layers of the Transformer, training an LLM from scratch. *K-Bert* [26], *KnowBert* [32], and *KP-PLM* [48] each propose new attention or injection layers, partially or fully overriding a model’s internal architecture. *K-Adapter* [50] appends adapters specialized in certain types of knowledge. While these methods effectively enhance LLMs with structured knowledge, most are tailored to encoder-only models (e.g., BERT) or require partial fine-tuning of the LLM. This complicates their usage in RAG deployments, where one may prefer to keep large-scale pretrained models intact to preserve existing capabilities. *ConceptFormer* aligns with these methods in its goal — bridging symbolic

graphs and neural representations — but preserves the fundamental architecture of a decoder-only LLM by operating strictly at the *input-embedding* level. In RAG scenarios with minimal inference memory or a preference for plug-and-play modules, *ConceptFormer* can be combined with a standard LLM in a non-invasive way. Furthermore, its *concept vectors* can be *precomputed* and stored for fast retrieval, eliminating inference overhead of reconstructing graph neighbors on the fly.

Gist Tuning and Prompt Compression. Gist tuning [31] condenses lengthy prompts into more compact “gist tokens.” Similar to knowledge textification, large prompt expansions can degrade performance or exceed context limits in tasks that rely on extensive background passages. By training a compressor to produce a short sequence of learnable tokens, gist tuning reduces the number of tokens needed, often maintaining high-quality generation. The idea of encoding complex information into a minimal embedding resonates with *ConceptFormer*, where entire KG neighborhoods are expressed as a few *concept vectors*. Instead of elaborating text expansions, *ConceptFormer* forms a high-level “gist” of an entity’s local subgraph.

Prompt Tuning and Continuous Prompts. Prompt tuning approaches [23, 27, 36] freeze most or all of an LLM’s parameters, introducing continuous embeddings (“prefix vectors” or “soft prompts”) that steer the model’s behavior. This differs from typical fine-tuning, which updates the model’s internal weights and can lead to catastrophic forgetting of original capabilities. *ConceptFormer* extends this paradigm: rather than encoding a generic style or instruction, it injects *structured, entity-centric knowledge*. Each KG node is accompanied by one or more learned embedding vectors that store local relational context. This is akin to “soft prompts” but specifically aimed at knowledge injection in a RAG setting.

Pseudo-Words for Specialized Concepts. A related vein of research addresses new concept acquisition in LLMs by inserting “pseudo words” into the model’s input space. In multimodal or domain-adaptation contexts, these pseudo tokens can represent, for instance, novel visual concepts [11, 44] or domain-specific terms. The idea is to attach new semantics to an otherwise unused token embedding, enabling the model to integrate or reference the concept during generation. *ConceptFormer* similarly represents new concepts (KG entities) via dense vectors, though it differs by focusing on subgraph-level knowledge, not just an image or a single domain label. By mapping an entire entity neighborhood into a few vectors, it preserves the topological integrity in a compressed embedding.

3 Datasets

Large Language Models generally learn from massive corpora that blend linguistic and factual content (e.g., Wikipedia), but such corpora alone are often suboptimal when the goal is to infuse *new* facts or systematically measure how effectively a model recalls or generates factual information in a controlled manner. To address this issue, we introduce three datasets—*T-REx Bite*, *Tri-REx*, and *T-REx Star*—that build upon the foundation of the *T-REx* dataset [9]. While *T-REx* links Wikipedia sentences to Wikidata triples, it was not designed specifically for next-token prediction. By contrast, our datasets provide explicit structures that facilitate both *knowledge*

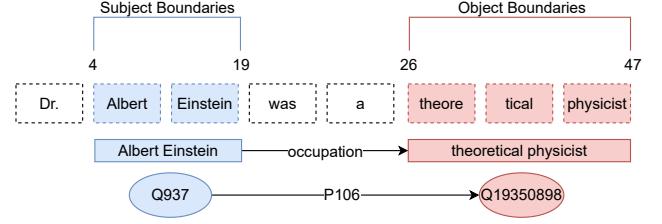


Figure 2: Example datapoint from *Tri-REx* (Synthetic) Dataset. The datapoint consists of the main sentence(s), information about the mentioned Wikidata triple, and boundary indications of the entity label locations.

injection and *retrieval-based* LLM evaluations, addressing key limitations of *T-REx* and its LAMA-based extensions [33, 34]. These limitations include the assumption of single-token objects, uneven coverage of knowledge, and inadequate alignment with the needs of decoder-only LLMs.

3.1 T-REx Bite

T-REx Bite [3] adapts *T-REx* to the next-token-prediction paradigm by ensuring that, in each text snippet, the subject appears before the object. This alignment mimics real-world scenarios in which a model sees partial information (the subject and some context) and must then predict the missing object. To keep snippets manageable within the limited context windows of smaller language models (such as GPT-2), each snippet is capped at 512 characters. We further require that the snippet explicitly mentions both subject and object, does not start a new sentence at the object, and is linked to the corresponding subgraph in *T-REx Star* (see Section 3.3).

By applying these constraints, we obtain about 6.4 million short “bites” for training, 0.92 million for testing, and 0.75 million for validation. Each bite is a compact piece of Wikipedia text that retains the original clarity and diversity of *T-REx* but is tailored to ensure a direct subject–object alignment. This structure lets researchers readily evaluate how well an LLM completes the object token(s) given the preceding subject. The dataset naturally accommodates multi-token objects under modern sub-word tokenization, removing the single-token assumptions of LAMA-like methods.

3.2 Tri-REx

Although *T-REx Bite* is useful, it still relies on Wikipedia text that many models may have partially seen during their pretraining phase. To create a scenario free from such potential contamination, we introduce *Tri-REx* [5]. Instead of extracting text from Wikipedia, *Tri-REx* synthesizes short subject–predicate–object sentences using Mistral 7B [17] in a few-shot prompting fashion. For example, a triple (*Albert Einstein, facial hair, walrus moustache*) might generate “Dr. Albert Einstein wore a bushy walrus moustache.” Each generated sentence is automatically filtered for coherence, correct mention of both subject and object, and accurate preservation of the S-P-O relationships, resulting in high-quality synthetic data. An illustration of a datapoint from *Tri-REx* is given in Figure 2.

Tri-REx comprises 21.5 million training sentences, 0.9 million test sentences, and 1.7 million validation sentences, each of which

is typically under 30 tokens. This collection stands out because it is intentionally *free of pretraining overlap*: models cannot simply rely on memorized Wikipedia text. Instead, they must learn or leverage newly provided knowledge sources (e.g., *concept vectors* from ConceptFormer) to recover the correct object tokens during next-token prediction. Researchers can thus verify whether a RAG technique or knowledge-injection approach genuinely grounds a model in facts, rather than triggering recall of memorized text.

3.3 T-REx Star

While T-REx Bite and Tri-REx focus on textual input, they do not explicitly embed the wider *graph structure* that connects subject entities to their neighbors. *T-REx Star* [4] fills this gap by providing star-topology subgraphs from Wikidata for each entity that appears as a subject in T-REx. Each entity’s local subgraph is represented in JSON format, includes up to 100 neighbors ranked by PageRank [42], and stores both node (Q-ID, English label, PageRank) and edge (P-ID, relation label) metadata. The JSON structure is easily loaded into tools such as NetworkX [15], enabling further graph-based processing or embedding.

Crucially, T-REx Star aligns with T-REx Bite and Tri-REx by using a consistent partitioning scheme. Every entity that serves as a *subject* in one of the three datasets appears in exactly one split (train, validation, or test). Entities may nonetheless appear as *objects* in multiple splits if they are neighbors of different subjects. This consistency is important for fair comparisons of LLM performance across training and evaluation sets.

4 Method

ConceptFormer is designed to inject structured, graph-native knowledge into a Large Language Model (LLM) without altering the LLM’s internal architecture. By operating purely at the *input embedding* level, *ConceptFormer* seamlessly integrates with most decoder-only LLMs and can be flexibly applied in retrieval-augmented generation, entity-centric search, or domain-specific IR pipelines. At a high level, *ConceptFormer* can be viewed as a *modular knowledge injector*:

- (1) **Entity Detection and Linking:** Given a user query or partial text (e.g., “Albert Einstein was a...”), an off-the-shelf Named Entity Recognition (NER) and entity linker identifies “Albert Einstein” and retrieves its corresponding node ID (Q_ID) in Wikidata or any other KG.
- (2) **Subgraph Extraction:** A star topology subgraph around the entity is fetched, containing its immediate neighbors and the edges (predicates) connecting them. In our experiments, subgraphs are pre-extracted from *T-REx Star*.
- (3) **ConceptFormer Vector Generation:** The extracted node and edge embeddings are fed into *ConceptFormer*, yielding *concept vectors* that capture the local neighborhood.
- (4) **Prompt Extension:** These *concept vectors* are appended to the existing input embeddings of the subject, forming a graph-grounded representation that the LLM processes natively (Figure 3).

This procedure allows an LLM to incorporate structured graph knowledge with minimal prompt overhead. Unlike text-based RAG approaches that rely on lossy linearization of graph structures

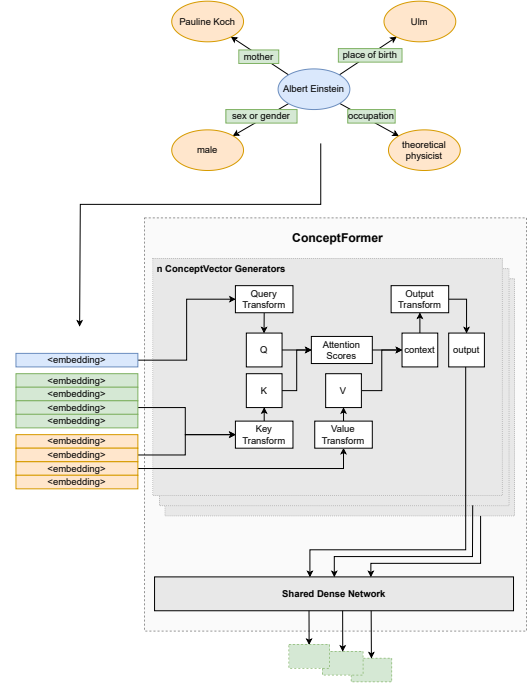


Figure 3: The input of the *ConceptFormer* are three matrices, representing the central node, neighbouring nodes, and connecting edges. These embeddings can be generated with numerous text-embedding mechanisms. *ConceptFormer* trains multiple, parallel, and fully independent *concept vector generator blocks*, each implementing an attention mechanism in which the central node becomes the query *Q*, the concatenated neighbouring nodes and corresponding edges become the key *K*, and the neighbouring nodes become the values *V*. A shared dense network transforms the output of each *concept vector generator block* into the input embedding space of the LLM.

(sometimes spanning hundreds of tokens), *ConceptFormer* only inserts n vectors per entity—where n is typically far smaller (e.g., 1 to 20 vectors). As a result, it preserves topological integrity while freeing up tokens for user text or system instructions in real-world RAG scenarios.

4.1 Architecture of ConceptFormer

ConceptFormer takes as input a *star topology subgraph* anchored on a central entity, which we denote as C . That entity has m neighbors $N_1, \dots, N_m$ and corresponding edge (predicate) embeddings $E_1, \dots, E_m$. Formally, each neighbor N_i is a fixed-dimensional vector (e.g., 768 D) representing the neighbor’s label or attributes, and each edge E_i is similarly embedded. These embeddings can be derived using off-the-shelf text embedding models (e.g., GPT-2, Word2Vec) or specialized methods like TransE [6] or PBG [19]. While our experiments focus on RDF-style triples, this architecture naturally extends to Property Graphs, where internal node attributes or edge properties can be encoded into the N and E vectors before injection.

In line with Liang et al. [24], we separate (i) the **node/edge embedding** step from (ii) the **alignment and compression** step. The former yields C, N, E with dimension $\dim_i$, and the latter is where *ConceptFormer*’s trainable parameters reside.

ConceptFormer implements n parallel concept vector generator blocks. Each of these generators learns a Key (K_n), Query (Q_n), Value (V_n), and Output (O_n) transformation through linear layers with weights W_n^Q, W_n^K, W_n^V , and W_n^O . The output of each concept attention layer is an intermediate *concept vector* ($1, \dim_i$), see Equations 1-4. Each O_n is then processed through an output transformation shared by all concept attention generators to produce the final *concept vectors*, see Equation 5. This transformation is critical for converting *concept vectors* into a format compatible with the LLM’s input space, a technique also employed in other papers to bridge the gap between symbolic graph representations like TransE and the LLM’s latent space [57]. The output dimension $\dim_o$ of this layer is defined by the input token embedding used by the LLM. Its hidden dimension is a free parameter, which we set to 1228 in our experiments.

$$Q_n = CW_n^Q \quad (1)$$

$$K_n = NW_n^K + E \quad (2)$$

$$V_n = NW_n^V \quad (3)$$

$$O_n = \left(\text{softmax} \left(\frac{Q_n K_n^T}{\sqrt{\dim_i}} \right) V_n \right) W_n^O \quad (4)$$

$$\text{concept vector}_n = \text{LeakyReLU}(O_n W_1^P) W_2^P \quad (5)$$

ConceptFormer can be fine-tuned to adapt to a new LLM, making it a versatile tool that can be integrated with various language models. The output of *ConceptFormer* is a set of n *concept vectors*, forming a matrix of size $(n, \dim_o)$. This matrix represents the transformed knowledge from the input subgraph, ready to be fed into the LLM for enriched language generation.

4.2 Training Objectives and Stages

We first train *ConceptFormer* on synthetic sentences from *Tri-REx*, which are deliberately *unseen* by the base LLM. This ensures that the LLM cannot trivially predict the correct object from memorized textual patterns. In other words, it must rely on the knowledge embedded in the *concept vectors*. For each synthetic sentence of the form “[Subject] ... [Predicate] ... [Object]”:

- (1) We truncate the sentence right before the [Object].
- (2) We retrieve the star subgraph of the [Subject] from *T-REx Star*, generate n concept vectors using *ConceptFormer*, and insert them after the original [Subject] text embedding vectors.
- (3) The LLM is tasked with next-token prediction of the [Object], minimizing the cross-entropy loss over the ground-truth tokens.

Since the LLM is frozen, gradient updates only affect *ConceptFormer* parameters. Over repeated batches, *ConceptFormer* learns to encode the subgraph in a way that helps the LLM correctly predict the object tokens, even though the LLM itself has not been fine-tuned on these unseen facts.

After pre-training, we further refine *ConceptFormer* using real-world text from *T-Rex Bite*, ensuring it transfers from the synthetic domain to more natural, context-rich sentences. This second stage is crucial to avoid an over-simplified reliance on “triplet schemas” alone. Concretely: (1) We apply the same next-token prediction objective on truncated real sentences from Wikipedia, referencing the same entity subgraphs. (2) The *concept vectors* must adapt to the varied, noisier style of genuine Wikipedia text rather than the neatly structured synthetic statements.

This two-stage training (synthetic $\rightarrow$ real text) has proved effective in our experiments, allowing *ConceptFormer* to handle both contrived and authentic RAG use cases—ranging from short fact queries to more elaborate, context-dependent queries.

4.3 Implementation Details and Training Hyperparameters

In our experiments, each concept vector generator block uses linear transformations of dimension $\dim_i \rightarrow \dim_i$, with $\dim_i$ typically 768. The subsequent MLP has one hidden layer of size 1228, using a LeakyReLU activation. We vary the number of parallel blocks $n \in \{1, 2, 3, 4, 5, 10, 15, 20\}$, balancing performance and memory overhead. We adopt AdamW with weight decay of 0.01 and a constant learning rate, typically 6×10^{-5} , determined via grid or Bayesian hyperparameter search. We freeze all LLM weights, ensuring stable optimization for *ConceptFormer* alone.

We batch multiple subgraphs and partial sentences, limiting each to a maximum token length (e.g., 512) for GPT-2 compatibility. We early-stop if validation loss does not improve after one full epoch in either the synthetic or real-data stage, preventing overfitting to narrower patterns. Once training is complete, *ConceptFormer* can be used in two ways: (1) **On-the-Fly Generation**: For each entity mention encountered during inference, dynamically retrieve its subgraph and run it through *ConceptFormer* to produce *concept vectors*. This supports real-time updates if the KG changes frequently but requires additional inference computation. (2) **Precomputed Lookup Table**: For static KGs like a fixed snapshot of Wikidata, generate *concept vectors* for all entities offline and store them in a giant key-value map. During inference, simply fetch the relevant *concept vectors* for each detected entity, incurring near-zero overhead.

The second approach is often attractive in Web-scale systems with stable or slow-changing knowledge bases, as it avoids re-running *ConceptFormer* repeatedly for the same entities.

5 Experiments

In this section, we investigate how well a Large Language Model can ground generation in facts from a knowledge graph, with a particular focus on Wikidata. By isolating knowledge-intensive tasks, we aim to show how *ConceptFormer* enables graph-native knowledge integration without exceeding context budgets or re-training the LLM.

5.1 Evaluation Paradigm and Metrics

We base our experiments on the *Tri-REx* and *T-REx Bite* datasets introduced in Section 3. Both sets consist of sentences truncated just before the mention of a target object entity (e.g., “Albert Einstein

was a...”), and the LLM is tasked with predicting the next tokens corresponding to the true entity label.

To quantify recall, we adopt the widely used Hit@k metric. For an object label split into T tokens, we record the rank (r_t) of each token t in the model’s output logits. The sequence rank is taken as $r = \max\{r_1, \dots, r_T\}$, and it counts as a “hit” if $r \leq k$ (i.e., all tokens appear in the top- k predictions at their respective timesteps). This approach is robust to multi-token entities, a common challenge in tasks involving named entities (“New York Times” vs. “NYT”).

Restricting to Hit@1 (top-1 predictions) may understate the model’s factual knowledge, since many facts can be phrased in multiple correct ways. Hence we primarily highlight Hit@10, though we report Hit@1 and Hit@5 for completeness.

5.2 Baseline Evaluation

We begin by establishing baseline performance for six LLMs of varying sizes and architectures: **GPT-2** (0.1B, 0.3B, 0.7B, 1.5B) [37] and **LLaMA-2** (3B, 7B) [43]. Each model is evaluated *as is*, without any knowledge injection. We measure Hit@1, Hit@5, and Hit@10 on the test splits of *Tri-REx* and *T-REx Bite*.

Table 1 (rows without RAG or CF) shows that larger models consistently outperform their smaller counterparts across both datasets, reflecting well-known scaling trends [18]. The gap between *Tri-REx* and *T-REx Bite* is particularly striking at Hit@1: performance on the Wikipedia-based *T-REx Bite* is often 5–10× higher than on the synthetic *Tri-REx*, suggesting that the model leverages memorized textual patterns from its pretraining corpus. This phenomenon highlights a key challenge: grounding generation beyond memorized facts to new or rare knowledge.

Moreover, LLaMA-2 variants (3B and 7B) display notably better results than GPT-2 models of even larger parameter counts (e.g., GPT-2 1.5B), implying that architecture and training methodology can greatly influence factual recall in IR tasks.

5.3 Baseline: Linearized Graph RAG

We next compare *ConceptFormer* to a text-based retrieval-augmented generation (RAG) approach. Specifically, for each subject entity, we retrieve its 1-hop neighbors from Wikidata and *linearize* them into a textual passage appended to the LLM prompt [10, 22].

We use a simple template-based approach to convert a graph neighbourhood to text. The performance varied significantly depending on the injection template used, sometimes leading to over a 100% difference in outcomes, consistent with findings from [27]. We observed that using a template in the form “Subject ({predicate}_1: {object}_1), {predicate}_2: {object}_2, ...)” performed particularly bad, while “Subject, {predicate}_1 {object}_1, {predicate}_2 {object}_2, ...”, a template also used by [8], performed best. However, this can easily span 100–800 tokens for well-known entities, obscuring structural relationships and consuming a significant portion of the LLM’s context window.

In Table 1, rows labeled “+ RAG” show large gains relative to the baseline. For smaller GPT-2 models, these gains can exceed 6× at Hit@10. This underscores the potential of structured knowledge for grounding if it is integrated effectively. However, the token overhead is substantial (on average 130 tokens per subject, but up to 800 for famous concepts), making it impractical in scenarios

Table 1: Percentage of correctly predicted entities per dataset of different models using no augmentation, linearized graph injection, or using different *ConceptFormer-n* variants.

Model	Tri-REx			T-REx Bite		
	H@1	H@5	H@10	H@1	H@5	H@10
LLaMA-2 7B	4.1%	17.5%	24.5%	39.3%	65.3%	73.0%
LLaMA-2 3B	4.3%	16.4%	22.9%	34.8%	59.5%	67.5%
GPT-2 1.5B	1.7%	8.8%	12.7%	22.9%	43.8%	51.8%
GPT-2 0.7B	1.6%	8.8%	12.4%	19.7%	39.3%	47.1%
GPT-2 0.3B	0.9%	7.0%	10.3%	16.3%	36.2%	44.2%
GPT-2 0.1B	1.3%	5.8%	8.5%	4.7%	14.3%	19.5%
LLaMA-2 7B + RAG	25.6%	61.0%	72.2%	55.3%	85.1%	90.6%
LLaMA-2 3B + RAG	28.4%	61.5%	71.7%	52.0%	82.3%	88.4%
GPT-2 1.5B + RAG	26.0%	54.2%	63.8%	39.6%	70.3%	78.3%
GPT-2 0.7B + RAG	20.9%	48.9%	59.1%	30.9%	60.0%	69.3%
GPT-2 0.3B + RAG	21.5%	50.7%	59.9%	30.5%	62.3%	71.4%
GPT-2 0.1B + RAG	8.3%	32.1%	41.3%	6.6%	23.8%	32.4%
GPT-2 0.1B + CF-20	16.8%	31.2%	36.9%	46.1%	65.6%	70.7%
GPT-2 0.1B + CF-15	16.2%	32.3%	38.2%	46.7%	67.1%	72.5%
GPT-2 0.1B + CF-10	15.7%	31.6%	37.8%	46.4%	67.2%	72.9%
GPT-2 0.1B + CF-5	13.6%	28.6%	35.1%	42.1%	63.3%	69.3%
GPT-2 0.1B + CF-4	12.7%	27.1%	33.7%	40.1%	61.5%	68.0%
GPT-2 0.1B + CF-3	11.9%	26.6%	32.9%	40.4%	61.0%	67.1%
GPT-2 0.1B + CF-2	11.1%	25.3%	31.7%	37.6%	57.8%	64.1%
GPT-2 0.1B + CF-1	10.0%	23.2%	28.8%	33.3%	54.1%	61.1%

where multiple enriched entities appear in a single query. We also find that performance can degrade for large neighborhoods due to linearization noise [26], in line with prior findings that excessive context can overwhelm the model.

5.4 ConceptFormer Evaluation

ConceptFormer takes a graph-native vector-based approach to knowledge integration, drastically reducing the number of extra tokens needed for an entity’s subgraph. We instantiate *ConceptFormer* with GPT-2 0.1B (125M parameters) to test whether a small model’s lack of knowledge can be compensated by structured prompts in *concept-vector* format. We employ the two-stage training scheme introduced in Section 4:

- (1) **Pre-training on Tri-REx:** We freeze GPT-2 0.1B and optimize *ConceptFormer* to generate the correct next tokens for synthetic subject-predicate-object sentences. This fosters reliance on external subgraph data, since the model cannot simply recall them from memorization.
- (2) **Fine-tuning on T-REx Bite:** We continue training *ConceptFormer* on real Wikipedia sentences. This step ensures the approach generalizes to authentic textual contexts, not just the minimal triplets from the synthetic set.

To explore the trade-off between vector capacity and prompt overhead, we train *ConceptFormer* variants with $n \in \{1, 2, 3, 4, 5, 10, 15, 20\}$. Each variant sees the same star subgraphs, but it can produce more or fewer concept vectors.

Table 1 (rows labeled “GPT-2 0.1B + CF- n ”) and Figure 4 illustrate that going from $n = 1$ to $n = 15$ markedly improves Hit@1 and Hit@10. Beyond 15, we see diminishing or no returns, suggesting that around 10–15 vectors are enough to cover typical 1-hop

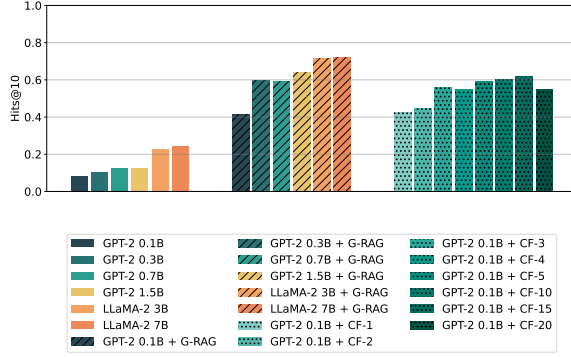


Figure 4: Hit@10 rate of various base models, with or without graph RAG (G-RAG), compared to GPT-2 0.1B with different *ConceptFormers* (CF), after pre-training on *Tri-REx*.

neighborhoods in *Tri-REx*. For GPT-2 0.1B, certain CF- n models even outperform LLaMA-2 7B, a 50 $\times$ larger model, in Hit@1—a striking result indicating that *concept vectors* can encode essential knowledge more compactly than the model’s own parameters.

After fine-tuning on real sentences, Table 1 shows the final performance. Notably, CF-15 yields a Hit@1 of 46.7% and Hit@10 of 72.5%—about a 10 $\times$ gain over baseline GPT-2 0.1B. Even a single concept vector ($n = 1$) outperforms text-based RAG for GPT-2 0.1B at Hit@1 (33.3% vs. 6.6%), while consuming 130 $\times$ fewer tokens on average. Figure 5 highlights the trade-off: graph textification saturates the input context, particularly for well-known subjects, whereas *ConceptFormer* retains high accuracy with minimal vector overhead. This is pivotal for Web-scale scenarios where multiple enriched entities may appear simultaneously.

Following Liu et al. [26], we partition subjects into:

- **Niche Concepts (1–10 neighbors):** Often less-known entities. CF-15 significantly outperforms RAG here, likely because text expansions for these entities are short, but the LLM still benefits from the learned concept representation.
- **Moderately Popular (11–90 neighbors):** Mixed results show consistent improvement for CF-15 over baseline GPT-2, though RAG also achieves good performance with moderate amounts of textual expansions.
- **Very Famous (90–100 neighbors):** Entities like “Albert Einstein” or “Queen Elizabeth II” can produce extremely large text expansions. RAG’s performance degrades, while CF-15 remains stable, illustrating that *ConceptFormer* avoids linearization noise issues when dealing with broad subgraphs.

Overall, *ConceptFormer* maintains high recall across varying entity degrees, making it attractive for tasks spanning rare to well-known topics.

5.5 Question Answering on WebQSP

To further validate *ConceptFormer* in a knowledge-grounded question answering context, we evaluate it on the WebQuestions Semantic Parsing (WebQSP) dataset [52]. Since WebQSP originally references Freebase, we use a Wikidata-linked variant [39]. After filtering questions with missing or incompatible subgraphs, we

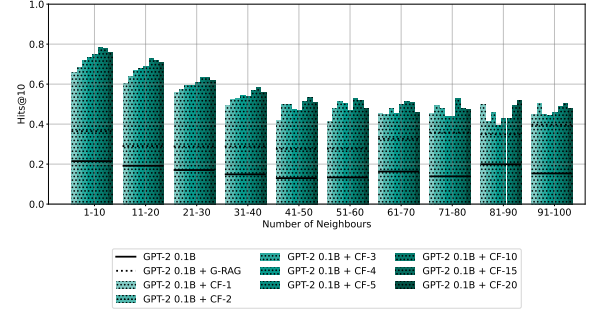


Figure 5: Hit@10 rate of GPT-2 0.1B + various *ConceptFormers* on the Wikipedia based *T-Rex Bite* Dataset.

form 2,463 question–answer pairs, each labeled with the relevant entity’s node ID.

We adapt GPT-2 0.1B to a prompt template: “Question: [Q]? Answer:”, then measure whether the correct entity label emerges in the top- k logits. As many questions have multiple valid answers or synonyms, we adopt a *looser* Hit@5 threshold: if any correct label is in the top-5 tokens, we consider it a recall success.

Table 2 compares:

- **Baseline GPT-2 0.1B:** 0% Hit@1, effectively guessing randomly for these specialized QA prompts.
- **+ RAG (Graph Textification):** Gains up to 1.4% Hit@5 for GPT-2 0.1B, or 13.0% for GPT-2 1.5B, but still low absolute performance.
- **+ *ConceptFormer*-10 (CF-10):** Achieves 7.6% Hit@1 and 28.3% Hit@5, vastly surpassing the LLaMA-2 7B result of under 1% Hit@1.

While these numbers are modest compared to specialized QA systems that can reach 75%+ [49], they demonstrate that an off-the-shelf 0.1B-parameter LLM can become *graph-grounded* through *ConceptFormer*’s vector-based injection. This highlights a path for using small LLMs in domain-oriented tasks (e.g., specialized KBs) without re-training a massive model.

5.6 Multi-Hop and Global Alignments

Although our experiments focus on 1-hop subgraphs, *ConceptFormer* can, in principle, be extended to multi-hop neighborhoods. Similarly, using globally aligned node embeddings (e.g., from PBG [19]) conferred no consistent advantage over simpler text-based embeddings. We hypothesize that *ConceptFormer* itself internalizes enough global structure through repeated 1-hop exposures.

5.7 Summary of Findings

Across a spectrum of evaluation settings:

- **Even a single *concept vector*** ($n = 1$) can boost GPT-2 0.1B from near-zero performance to competitive results while consuming $\sim 100\times$ fewer tokens than linearized graph expansions.
- **15 *concept vectors*** emerges as an effective upper bound for 1-hop neighborhoods, pushing recall higher but still remaining token-efficient.

Table 2: Percentage of correctly answered questions from the WebQSP Dataset, comparison of the base models (BM), graph RAG (G-RAG), and *ConceptFormer-10* (CF-10).

Model	BM		G-RAG		CF-10	
	H@1	H@5	H@1	H@5	H@1	H@5
LLaMA-2 7B	0.9%	13.5%	0.1%	6.1%		
LLaMA-2 3B	0.1%	10.3%	0.0%	10.1%		
GPT-2 1.5B	0.1%	4.2%	0.2%	13.0%		
GPT-2 0.7B	0.0%	3.9%	0.1%	2.4%		
GPT-2 0.3B	0.0%	1.5%	0.0%	1.1%		
GPT-2 0.1B	0.0%	0.0%	0.2%	1.4%	7.6%	28.3%

- **Complex or Large Subgraphs** that hamper text-based RAG due to linearization noise have less impact on *ConceptFormer*, which compresses the subgraph into a fixed number of vectors.
- **Knowledge-Grounded QA** scenarios like WebQSP show that *ConceptFormer* can transform a small LLM into a basic QA engine for domain knowledge queries. Performance, while not state-of-the-art, underscores the viability of vector-based knowledge injection.

Overall, our experiments confirm that *ConceptFormer* provides robust, scalable means to ground generation in structured knowledge with minimal overhead—a boon for tasks where preserving topological integrity within tight context windows is critical. We next conclude by discussing limitations, future work, and broader implications for retrieval-augmented generation.

6 Conclusion

This paper presented *ConceptFormer*, a novel approach to ground Large Language Models in structured knowledge from Knowledge Graphs without modifying their internal architecture. Our experiments demonstrated that *ConceptFormer* significantly enhances the factual reliability of GPT-2 0.1B. This improvement is attributed to the neuro-symbolic transformation of KG information into compact and informative *concept vectors*. The method of creating and injecting *concept vectors* into the LLM input space offers a powerful way to enrich LLMs with current and detailed world knowledge while preserving latent capacity for user queries or other contextual prompts.

ConceptFormer provides a solution for knowledge-intensive tasks such as query expansion, entity-centric retrieval, and knowledge-grounded question answering. By embedding graph information at the input-embedding level rather than via lossy textual linearization, *ConceptFormer* reduces the risk of linearization noise—making it highly scalable for multi-entity queries often seen in advanced retrieval workflows. It bridges the gap between neural generation and symbolic structures, aligning with the broader shift toward trustworthy neuro-symbolic AI.

Our results show that *ConceptFormer* achieves superior performance compared to raw LLMs and even outperforms template-based graph RAG methods in most scenarios (cf. Table 3). Notably, it enhances knowledge recall with a minimal increase in context size,

Table 3: Performance change of GPT-2 0.1B + 15 *concept vectors*, compared to GPT-2 0.1B base model and GPT-2 0.1B + graph RAG (G-RAG).

Model	Tri-REx			T-Rex Bite		
	H@1	H@5	H@10	H@1	H@5	H@10
GPT-2 0.1B	1121% ↑	457% ↑	348% ↑	894% ↑	370% ↑	271% ↑
GPT-2 0.1B + G-RAG	93% ↑	0% ≈	8% ↓	612% ↑	181% ↑	123% ↑

providing an efficient pathway for integrating large-scale knowledge bases into modern IR pipelines. The effectiveness of *ConceptFormer* is especially notable when comparing the input token usage between *ConceptFormer-1* and graph RAG: just one singular *concept vector* can outperform graph RAG on the *T-Rex Bite* dataset by 88% on Hit@10. Such savings directly benefit tasks that require injecting relevant knowledge for multiple entities within a single prompt, including complex queries or conversation-based searches.

Once trained, *ConceptFormer* can pre-generate a comprehensive lookup table that maps entities to *concept vectors*. Alternatively, it can be used dynamically, querying the relevant neighborhood from the source graph on the fly and embedding it into input space-compatible *concept vectors*, thus offering a streamlined retrieval-augmentation pipeline. Changes in the online KG are automatically reflected and made accessible to the LLM, making *ConceptFormer* suitable for highly dynamic retrieval scenarios with fast-changing graphs. Overall, it provides a versatile, graph-native bridge between the structured world of KGs and the generative capabilities of LLMs, supporting more robust and up-to-date information retrieval. *ConceptFormer* exhibits four key properties: (1) **Topological Fidelity**: Unlike text-based RAG which flattens graphs into sequences, our vector injection preserves the relational structure of the neighborhood, reducing linearization noise. (2) **Representational Efficiency**: Each entity’s neighborhood is compressed into a handful of vectors ($\approx 1\text{--}20$ “soft tokens”) compared to hundreds of tokens for text-based RAG, saving crucial context budget. (3) **No Fine-tuning**: By remaining purely in the input space, it allows reusing standard models, which is valuable where re-training is infeasible or restricted. (4) **Property Graph Compatibility**: The method is agnostic to the embedding technique and naturally extends to Property Graphs, allowing the integration of specialized domain knowledge (e.g., medical or legal) with rich node/edge attributes.

These features underscore *ConceptFormer*’s suitability for retrieval-based generation, question answering, or any other task that demands up-to-date factual grounding. By prioritizing representational efficiency, modularity, and dynamic scalability, *ConceptFormer* fills a critical gap in bridging structured graph data with frozen Large Language Models while minimizing the burden on context budgets. Ultimately, it unifies structured graph knowledge with LLM-based generation in a manner that is practical, extensible, and highly aligned with the requirements of modern information retrieval systems.

Acknowledgments

This work was partially supported by the Swiss National Foundation through the project ‘MediaGraph’ (contract no. 202125).

References

- [1] Bilal Abu-Salih. 2021. Domain-specific knowledge graphs: A survey. *J. Netw. Comput. Appl.* 185 (2021), 103076. doi:10.1016/J.JNCA.2021.103076
- [2] Joel Barmettler, Abraham Bernstein, and Luca Rossetto. 2025. ConceptFormer trained on T-REx Lite. doi:10.5281/zenodo.15187984
- [3] Joel Barmettler, Abraham Bernstein, and Luca Rossetto. 2025. *T-REx Bite 1.0*. doi:10.5281/zenodo.15165883
- [4] Joel Barmettler, Abraham Bernstein, and Luca Rossetto. 2025. *T-REx Star 1.0*. doi:10.5281/zenodo.15165974
- [5] Joel Barmettler, Abraham Bernstein, and Luca Rossetto. 2025. *Tri-REx 1.0*. doi:10.5281/zenodo.15166163
- [6] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. 2787–2795. <https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html>
- [7] Ryan Brate, Minh Hoang Dang, Fabian Hoppe, Yuan He, Albert Meroño-Peñuela, and Vijay Sadashivaiah. 2022. Improving Language Model Predictions via Prompts Enriched with Knowledge Graphs. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022), Virtual Conference, on-line, October 24, 2022 (CEUR Workshop Proceedings, Vol. 3342)*. CEUR-WS.org. <https://ceur-ws.org/Vol-3342/paper-3.pdf>
- [8] Ryan Brate, Minh Hoang Dang, Fabian Hoppe, Yuan He, Albert Meroño-Peñuela, and Vijay Sadashivaiah. 2022. Improving Language Model Predictions via Prompts Enriched with Knowledge Graphs. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022), Virtual Conference, on-line, October 24, 2022 (CEUR Workshop Proceedings, Vol. 3342)*. CEUR-WS.org. <https://ceur-ws.org/Vol-3342/paper-3.pdf>
- [9] Hady Elsahar, Pavlos Vougiouklis, Arslan Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2018/summaries/632.html>
- [10] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a Graph: Encoding Graphs for Large Language Models. *CoRR abs/2310.04560* (2023). arXiv:2310.04560 doi:10.48550/ARXIV.2310.04560
- [11] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit Haim Bermano, Gal Chechik, and Daniel Cohen-Or. 2023. An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. <https://openreview.net/pdf?id=NAQvF08TcyG>
- [12] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-Augmented Generation for Large Language Models: A Survey. *CoRR abs/2312.10997* (2023). arXiv:2312.10997 doi:10.48550/ARXIV.2312.10997
- [13] 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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, 179–188. doi:10.18653/V1/P17-1017
- [14] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks Are All You Need. *CoRR abs/2306.11644* (2023). arXiv:2306.11644 doi:10.48550/ARXIV.2306.11644
- [15] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*. Pasadena, CA USA, 11 – 15.
- [16] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Trans. Neural Networks Learn. Syst.* 33, 2 (2022), 494–514. doi:10.1109/TNNLS.2021.3070843
- [17] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Léo Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *CoRR abs/2310.06825* (2023). arXiv:2310.06825 doi:10.48550/ARXIV.2310.06825
- [18] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale Fung, Mohammad Shoeybi, and Bryan Catanzaro. 2022. Factuality Enhanced Language Models for Open-Ended Text Generation. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. http://papers.nips.cc/paper_files/paper/2022/hash/df438caa36714f69277daa92d608dd63-Abstract-Conference.html
- [19] Adam Lerer, Ledell Wu, Jiajun Shen, Timothée Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. 2019. Pytorch-BigGraph: A Large Scale Graph Embedding System. In *Proceedings of Machine Learning and Systems 2019, MLSys 2019, Stanford, CA, USA, March 31 - April 2, 2019*. mlsys.org. <https://proceedings.mlsys.org/book/282.pdf>
- [20] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html>
- [21] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained Language Models for Text Generation: A Survey. *CoRR abs/2105.10311* (2021). arXiv:2105.10311 <https://arxiv.org/abs/2105.10311>
- [22] Shiyang Li, Yifan Gao, Haoming Jiang, Qingyu Yin, Zheng Li, Xifeng Yan, Chao Zhang, and Bing Yin. 2023. Graph Reasoning for Question Answering with Triplet Retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 3366–3375. doi:10.18653/V1/2023.FINDINGS-ACL.208
- [23] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 4582–4597. doi:10.18653/V1/2021.ACL-LONG.353
- [24] Sheng Liang, Mengjie Zhao, and Hinrich Schütze. 2022. Modular and Parameter-Efficient Multimodal Fusion with Prompting. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 2976–2985. doi:10.18653/V1/2022.FINDINGS-ACL.234
- [25] Adam Liska, Tomáš Kociský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d’Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsonan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. StreamingQA: A Benchmark for Adaptation to New Knowledge over Time in Question Answering Models. In *International Conference on Machine Learning, ICLR 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 13604–13622. <https://proceedings.mlr.press/v162/liska22a.html>
- [26] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: Enabling Language Representation with Knowledge Graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2901–2908. doi:10.1609/AAAI.V34I03.5681
- [27] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT Understands, Too. *CoRR abs/2103.10385* (2021). arXiv:2103.10385 <https://arxiv.org/abs/2103.10385>
- [28] Kelvin Luu, Daniel Khashabi, Suchin Gururangan, Karishma Mandyam, and Noah A. Smith. 2022. Time Waits for No One! Analysis and Challenges of Temporal Misalignment. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. Association for Computational Linguistics, 5944–5958. doi:10.18653/V1/2022.NAACL-MAIN.435
- [29] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FactScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 12076–12100. <https://aclanthology.org/2023.emnlp-main.741>
- [30] Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha P. Talukdar, Bo Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matt Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Nandapaula Nakashole, Emmanouil A. Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. 2018. Never-ending learning. *Commun. ACM* 61, 5 (2018), 103–115. doi:10.1145/3191513
- [31] Jesse Mu, Xiang Li, and Noah D. Goodman. 2023. Learning to Compress Prompts with Gist Tokens. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New*

- Orleans, LA, USA, December 10 - 16, 2023. http://papers.nips.cc/paper_files/paper/2023/hash/3d77c6dc7f143aa2154e7f4d5e22d68-Abstract-Conference.html
- [32] Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. 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 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 43–54. doi:10.18653/V1/D19-1005
- [33] Fabio Petroni, Patrick S. H. Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How Context Affects Language Models' Factual Predictions. In *Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020*. doi:10.24432/C5201W
- [34] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language Models as Knowledge Bases?. 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 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 2463–2473. doi:10.18653/V1/D19-1250
- [35] Nina Pörner, Ulli Waltinger, and Hinrich Schütze. 2019. BERT is Not a Knowledge Base (Yet): Factual Knowledge vs. Name-Based Reasoning in Unsupervised QA. *CoRR* abs/1911.03681 (2019). arXiv:1911.03681 <http://arxiv.org/abs/1911.03681>
- [36] Guanghui Qin and Jason Eisner. 2021. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*. Association for Computational Linguistics, 5203–5212. doi:10.18653/V1/2021.NAACL-MAIN.410
- [37] Alec Radford, Jeff Wu, Rewon Child, D. Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. <https://www.semanticscholar.org/paper/Language-Models-are-Unsupervised-Multitask-Learners-Radford-Wu/9405cc0d6169988371b2755e573cc28650d14dfe>
- [38] Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How Much Knowledge Can You Pack Into the Parameters of a Language Model?. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 5418–5426. doi:10.18653/V1/2020.EMNLP-MAIN.437
- [39] Daniil Sorokin and Iryna Gurevych. 2018. Modeling Semantics with Gated Graph Neural Networks for Knowledge Base Question Answering. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. Association for Computational Linguistics, 3306–3317. <https://aclanthology.org/C18-1280/>
- [40] Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. 2023. Head-to-Tail: How Knowledgeable are Large Language Models (LLMs)? A.K.A. Will LLMs Replace Knowledge Graphs? *CoRR* abs/2308.10168 (2023). arXiv:2308.10168 doi:10.48550/ARXIV.2308.10168
- [41] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*. International Committee on Computational Linguistics, 3660–3670. doi:10.18653/V1/2020.COLING-MAIN.327
- [42] Andreas Thalhammer and Achim Rettinger. 2016. PageRank on Wikipedia: Towards General Importance Scores for Entities. In *Joint Proceedings of the 5th Workshop on Data Mining and Knowledge Discovery meets Linked Open Data and the 1st International Workshop on Completing and Debugging the Semantic Web (Know@LOD-2016, CoDeS-2016) co-located with 13th ESWC 2016, Heraklion, Greece, May 30th, 2016 (CEUR Workshop Proceedings, Vol. 1586)*. CEUR-WS.org. <https://ceur-ws.org/Vol-1586/know2.pdf>
- [43] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajwal Bhargava, Shrutti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *CoRR* abs/2307.09288 (2023). arXiv:2307.09288 doi:10.48550/ARXIV.2307.09288
- [44] Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal Few-Shot Learning with Frozen Language Models. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. 200–212. <https://proceedings.neurips.cc/paper/2021/hash/01b7575c38dac42f3cfb7d500438b875-Abstract.html>
- [45] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85. doi:10.1145/2629489
- [46] Hongmin Wang. 2020. Revisiting Challenges in Data-to-Text Generation with Fact Grounding. *CoRR* abs/2001.03830 (2020). arXiv:2001.03830 <https://arxiv.org/abs/2001.03830>
- [47] Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Haojun Huang, Wei Ye, Xiubo Geng, Binxing Jiao, Yue Zhang, and Xing Xie. 2023. On the Robustness of ChatGPT: An Adversarial and Out-of-distribution Perspective. *CoRR* abs/2302.12095 (2023). arXiv:2302.12095 doi:10.48550/ARXIV.2302.12095
- [48] Jianing Wang, Wenkang Huang, Minghui Qiu, Qiuhui Shi, Hongbin Wang, Xiang Li, and Ming Gao. 2022. Knowledge Prompting in Pre-trained Language Model for Natural Language Understanding. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 3164–3177. doi:10.18653/V1/2022.EMNLP-MAIN.207
- [49] Ruijie Wang, Luca Rossetto, Michael Cochez, and Abraham Bernstein. 2023. GNN2R: Weakly-Supervised Rationale-Providing Question Answering over Knowledge Graphs. *CoRR* abs/2312.02317 (2023). arXiv:2312.02317 doi:10.48550/ARXIV.2312.02317
- [50] Ruizhe Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021 (Findings of ACL, Vol. ACL/IJCNLP 2021)*. Association for Computational Linguistics, 1405–1418. doi:10.18653/V1/2021.FINDINGS-ACL.121
- [51] Weiqi Wu, Chengyue Jiang, Yong Jiang, Pengjun Xie, and Kewei Tu. 2023. Do PLMs Know and Understand Ontological Knowledge?. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 3080–3101. doi:10.18653/V1/2023.ACL-LONG.173
- [52] Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The Value of Semantic Parse Labeling for Knowledge Base Question Answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics. doi:10.18653/V1/P16-2033
- [53] Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2022. A Survey of Knowledge-enhanced Text Generation. *ACM Comput. Surv.* 54, 11s (2022), 227:1–227:38. doi:10.1145/3512467
- [54] Taolin Zhang, Chengyue Wang, Nan Hu, Minghui Qiu, Chengguang Tang, Xiaofeng He, and Jun Huang. 2022. DKPLM: Decomposable Knowledge-Enhanced Pre-trained Language Model for Natural Language Understanding. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelfth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 11703–11711. doi:10.1609/AAAI.V36I10.21425
- [55] Xinxin Zheng, Feihu Che, Jinyang Wu, Shuai Zhang, Shuai Nie, Kang Liu, and Jianhua Cao. 2024. KS-LLM: Knowledge Selection of Large Language Models with Evidence Document for Question Answering. arXiv:2404.15660 [cs.CL] <https://arxiv.org/abs/2404.15660>
- [56] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less Is More for Alignment. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*. http://papers.nips.cc/paper_files/paper/2023/hash/ac662d74829e4407ce1d126477f4a03a-Abstract-Conference.html
- [57] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 4623–4629. doi:10.24963/IJCAI.2018/643