My research focuses on artificial intelligence (AI) systems that can leverage arbitrarily long contexts with minimal trade-offs on efficiency and performance. Imagine a scenario where a reader who wants to skim over a novel, and a virtual assistant can quickly summarize the plot and answer the reader's follow-up questions by fetching evidence from the book. To build such a system, language models (LMs) need to comprehend complicated texts beyond paragraphs. Except for improving the efficiency of LMs on *long texts*, a crucial research problem is to guarantee they can *effectively* utilize them. Moreover, beyond the LMs themselves, an external database may be used to cache the information of processed texts, making the size of context nearly *unlimited* (Figure 1).

Challenges of LMs on long sequences The transformer model is inefficient for long sequences, and numerous transformer variants aim to improve its efficiency. I conduct controlled experiments to uncover the issues of these models: Though they scale up to long contexts, the utilization of distant information is not improved. Reducing the complexity may be a misleading way toward long-range transformers (Qin et al., 2023a).

Rethinking the atomic units of LMs Tokens (words) were considered as the atomic units for LMs, but my work represents texts in different resolutions, such as *segments*. My work first reveals that LMs can take vectors as input: Tokens can be replaced with "soft prompts" (Qin and Eisner, 2021). Beyond token-level embeddings, a paragraph can be represented with dynamic numbers of vectors (Qin and Van Durme, 2023; Qin et al., 2023b). In terms of efficient transformers, unlike prior work that tries to reduce the exponent in the $O(n^2)$ complexity of transformers, my research instead reduces the sequence length n.

Interacting with external database Even with efficient language representation, local context cannot accommodate infinite information. To store and access domain knowledge and past conversations, which might be of various modalities such as text, pictures, and audio, an advanced AI system ought to store them offline and "side-load" them with retrieval. I study the interaction between sequence models and temporal databases (Mei et al., 2020, 2019), where events are dynamically saved in a declarative database, which in turn affects future predictions.

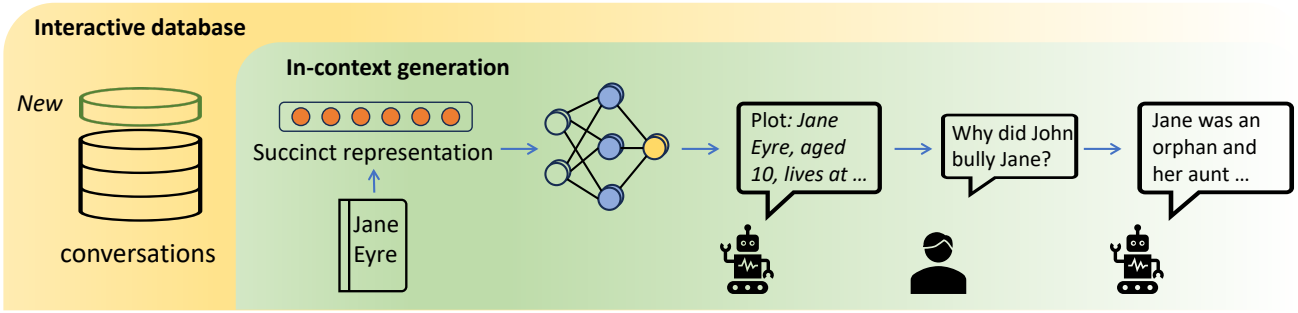

Figure 1: An AI system is capable of encoding a book succinctly with vectors much fewer than tokens. When asked follow-up questions, it retrieves evidence from the book representation. Past conversations are saved to a database and can be retrieved for future conversations.

1 Challenges of LMs on Long Sequences

Transformers are non-scalable to long contexts in nature for their quadratic computational complexity in both time and space. Therefore, many prior works aim to lower the complexity with efficient methods by novel designs on the self-attention of transformers. **However, their effectiveness in NLP tasks remains a question**: Most of them do not have massive pretraining on texts, and thus cannot be applied to downstream tasks. On the other hand, even with pretraining, controlled experiments are needed to verify the contribution of long-range attention.

The major challenge in analyzing long-range transformers is the discrepancy between their pretraining setups and model architectures. In my work Qin et al. (2023a), I designed experiments to isolate these confounders.

For cross-model comparison, *parameters are migrated from pretrained checkpoints to compensate for the lack of pretraining*. For the investigation of each architecture, *long-range attention is ablated* as controlled factors.

My research indicates that 3 major solutions for long-range attention could fail in different cases. Methods that sparsify attention patterns, e.g. Longformer (Beltagy et al., 2020), down-perform its short-range ablation in some cases. Recurrence-based transformers, e.g. TransformerXL (Dai et al., 2019), often fail to utilize the distant information. Kernel methods, e.g. Performer (Choromanski et al., 2021), though achieve up to linear complexity, suffer from error accumulation across transformer layers, which is translated into the lowest performance. It is astounding to observe the fragility in real NLP applications behind the prosperity of efficiency transformers. [1]

2 Rethinking the Atomic Units of LMs

Tokens are considered the *atomoc units* of LMs: Every token is encoded into one contextualized embedding vector. However, tokens are of different importance in a sequence, and treating them as uniform sequences does not allow us to flexibly allocate the compute. My work tries to answer: Can we feed LMs with non-token inputs that may contain richer information? Can we find a resolution for text coarser than tokens, like segment-level?

2.1 Talking to LMs with "Soft Prompts" (The Best Short Paper in NAACL 2021)

Prompting is a natural way to query LMs, and prompt engineering has become a popular subject in NLP research. However, the quality of prompts heavily affects their performance, and long prompts can be computationally costly. My research aims to automatically search for effective and shorter prompts.

Optimizing a prompt can be difficult because the tokens of natural language are in a discrete space. It can be much easier, however, if prompts are relaxed into *a continuous space*. In my work Qin and Eisner (2021), the tokens in prompts are replaced with *vectors*, called **soft prompts**. Soft prompts are fed into LMs by skipping the embedding layer, thus being trainable via back-propagation (Figure 2). Soft prompts provide a much larger space for optimization and can express semantics that are hard or even impossible with natural language. E.g. LMs are confused about the prompt "`Mary Cassatt died in __`" because both `1926` and `Paris` are valid answers, but soft prompts can replace the word `in` with the non-English vector `in`$_{\texttt{year}}$, which is a specialized preposition to guide the LM to generate a year to fill the blank.

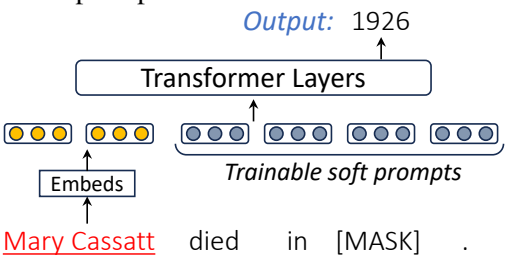

Figure 2: An example of soft prompts that replace the tokens "`died in [MASK] .`"

Soft prompts affect the embedding layer, and the hidden states of prompts in every layer of transformers can be trainable, called *deep perturbation*. Moreover, multiple soft prompts can form an ensemble, making it *a Mixture-of-Export (MoE) system*. Soft prompts, together with these tricks, strongly outperform the natural language prompt baselines. Our experiments suggest that the potential of LMs is far from being exploited.

2.2 Encoding Texts into "Nuggets": A Resolution Coarser than Tokens

One-vector-per-token is the de facto way of text representation in LMs. but this is counterfactual because the information can be non-uniformly distributed in texts. *I propose that the embedding resolution of LMs can be coarser than tokens*, seeking a "semantically useful level of granularity" (Rudinger et al., 2017). More specifically, can we find a method to encode texts with a dynamic number of vectors (Figure 3)?

[1] As a side observation, open-sourced large language models (LLMs) rarely use these methods even for long contexts.

My solution is called NUGGET, standing for Neural Agglomerative Embeddings of Text (Qin and Van Durme, 2023). In NUGGET, LMs learn to subselect a fractional number of token embeddings to represent texts. The selection process is not differentiable but can be learned through a residual connection. Based on an encoder-decoder transformer (Lewis et al., 2020a), NUGGET can be trained through objectives such as autoencoding and machine translation. I surprisingly found that 15% of token embeddings are already sufficient to produce nearly lossless encoding, suggesting a *segment-level resolution* can be practically useful without loss of information.

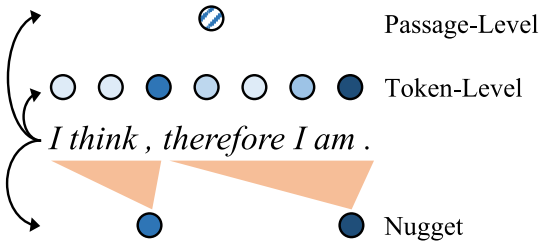

Figure 3: Instead of per-token or per-sequence, Nugget uses dynamic numbers of vectors.

The *intrinsic behavior* of Nugget is even more surprising. Without guidance during training, it learns to use clausal text delimiters, e.g. punctuations and conjunction words, to represent text. Inspecting these embeddings, each of them roughly encodes a contiguous segment of text preceding the delimiter. It implies a *divide-and-conquer* text representation solution is *spontaneously* learned by NUGGET.

The change of resolution also works for decoder-only LLMs. In my work Qin et al. (2023b), an LLM like LLaMA (Touvron et al., 2023) learns to encode a chunk of text with a dynamical number of vectors. The compression level is determined by the importance of the text, e.g. supporting documents are less important and can be compressed with a ratio up to 20x. Experiments show that **LLaMA can perfectly reconstruct the compressed documents with only 5% of the hidden states**, and the performance on downstream applications, such as question answering and summarization, can be mostly preserved with compression.

3 Sequence Models Interacting with Database

I also researched probabilistic sequence modeling for **temporal events**, where discrete events happen in a continuous timeline. Think of soccer game records that include events such as players passing or kicking the ball at certain timestamps. Our goal is to model the distribution of $p(\text{missing events} \mid \text{observed events})$.

In Mei et al. (2019), we consider modeling the probability of missing events given past and future observed events. Deriving this conditional distribution from complete event probability $p(\text{events})$ is intractable, thus we propose to use *particle smoothing* — a form of sequential importance sampling — to impute missing events. In the decoding phase, I propose an *consensus decoding* algorithm that can efficiently approximate the decode with minimum Bayes risk (MBR), which is shown to be more effective than maximum a posteriori (MAP) estimation.

In Mei et al. (2020), we study **the interaction between sequence models and a database**. The number of event types can be combinatorially large when *events are structured*, making event modeling difficult. In the soccer game example, there can be $\#(\text{players}) \times \#(\text{actions})$ events. Not all events are legal, e.g. only the player with the ball can goal, so we impose the domain knowledge to the model with a *neural-symbolic* system implemented with Datalog, a declarative logic language, as the database. New events are added to the database, and domain rules regulate the types of future events.

Our experiments on event sequence modeling suggest a way to extend the capability of LMs beyond neural networks: External database. Its benefit can be two-fold: On the one hand, the language generation process can be harnessed by domain knowledge, like the logic rules in Datalog. On the other hand, by interacting with a database, LMs can cache their past activations to out-of-context memories. We can consequently make the effective context length of LMs nearly unlimited.

4 Future Research Directions

Though LLMs have made rapid progress in natural language understanding and generalization, larger models do not asymptotically present human-level ability to comprehend long texts. Except for the dizzying number of transformer variants (Tay et al., 2022), the efficiency and performance of LMs on long sequences largely remains unsatisfactory, therefore their deployment remains expensive. The first step of my research plan is to *reduce the cost of the deployment of LLMs*. My research aims to reduce the cost of transformers **in modules other than self-attention**. My research will also delve into practical approaches towards efficient NLP with 2 prongs: *in-context* and *out-of-context*. For the in-context LMs, my research aims to **enable LMs to efficiently store contextual information**. This idea can then be extended to **out-of-context** scenarios, where domain-specific knowledge or past conversations are stored in *an external database* that stores information in the form of *vectors*.

Reducing the cost of the deployment of LLMs With the tremendous size of LLMs, the bottleneck of computation efficiency is the multi-layer perceptron (MLP) modules instead of the self-attention for common sequence lengths. Low-precision methods, such as quantization and low-rank approximation, suffer from performance loss because they evenly trade in the precision across all parameters. Instead, based on the importance of individual modules, one could **adaptively trade-off the precision and the performance** to maximize the utility of LLMs with a limited computational budget. Moreover, we could **learn to approximate** by dynamically controlling the precision of each dense matrix computation. Similar to the mixture of experts (MoE), we achieve "**soft MoE**" system with the approximation as our gates.

Language representation with arbitrary granularity Feeding text as uniform token sequences to LMs is wasteful: The "semantic density" of text is not necessarily proportional to its length, and the importance of each chunk of text may depend on its relationship with the target context. I have shown that text can be represented beyond the "one-vector-per-token" paradigm with more succinct vectors called NUGGET. My next research direction is to encode text with **mixed levels of granularity**, where LMs dynamically allocate compute to different parts of the context with different compression levels, exploiting the limited memory and compute. A follow-up problem is, can LMs encode text **hierarchically** with different abstractiveness at different levels? When generating new tokens, LMs can flexibly attend to past hidden states at different levels. This not only extends the context length but also biases the LMs to selectively attend to past texts. Imagine a novel is hierarchically encoded with different levels of granularity, and an LM can summarize its plot by attending to a few abstractive vector representations. When asked about details, it then fetches the lower-level representations of the corresponding texts to extract finer details.

Unbounded memory with external vector database and information retrieval The above research ideas are concerned with the contexts of LMs, but in-context information is eventually bounded by the memory. Therefore, I believe the prowess of LMs can be fully unleashed when equipped with *external database* that can store information such as domain knowledge and past conversations. Retrieval-augmented generation (RAG) (Lewis et al., 2020b) attempts to introduce additional information to LMs in the form of texts, but texts can be inefficient and less expressive. Instead, one could **save the text encodings in a vector database**. Together with hierarchical encoding, the vector database can be of arbitrary sizes. Another advantage of vectors over tokens is that **dense retrieval and context encoding can be jointly conducted**, greatly reducing the latency of RAG. A vector database greatly extends the context length of LMs, enabling a chatbot to have an *never-end* conversation with users. Looking into the future, a vector database can be more capable in terms of **multimodality**, where the model stores in the database any information that can be encoded into vectors, such as visual or acoustic features.

My Relevant Publications

Hongyuan Mei, Guanghui Qin, and Jason Eisner. 2019. Imputing Missing Events in Continuous-Time Event Streams. In *International Conference on Machine Learning (ICML)*.

Hongyuan Mei, Guanghui Qin, Minjie Xu, and Jason Eisner. 2020. Neural Datalog Through Time: Informed Temporal Modeling via Logical Specification. In *International Conference on Machine Learning (ICML)*.

Guanghui Qin and Jason Eisner. 2021. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Guanghui Qin, Yukun Feng, and Benjamin Van Durme. 2023a. The NLP Task Effectiveness of Long-Range Transformers. In *Annual Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Guanghui Qin, Corby Rosset, Ethan C. Chau, Nikhil Rao, and Benjamin Van Durme. 2023b. Nugget2D: Dynamic Contextual Compression for Scaling Decoder-only Language Models.

Guanghui Qin and Benjamin Van Durme. 2023. Nugget: Neural Agglomerative Embeddings of Text. In *International Conference on Machine Learning (ICML)*.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. 2021. Rethinking Attention with Performers. In *International Conference on Learning Representations (ICLR)*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Skip-Prop: Representing Sentences with One Vector Per Proposition. In *International Conference on Computational Semantics (IWCS)*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient Transformers: A Survey. *ACM Computing Surveys*, 55(6):1–28.

Hugo Touvron, Lavril Thibaut, and et. al. 2023. LLaMA: Open and Efficient Foundation Language Models.