Grammar-Constrained Decoding For Structured NLP Tasks without Fine-Tuning

Saibo Geng, Martin Josifoski, Maxime Peyrard*, Robert West
Data Science Lab@EPFL, Switzerland
*Université Grenoble Alpes, CNRS, Grenoble INP, LIG
saibo.geng@epfl.ch

EMNLP 2023

Motivation

- LLM struggles with generating **complex structured outputs** without specific **fine-tuning**, like **json** or **XML** documents.
- **Grammar-Constrained Decoding (GCD)** uses **formal grammars** and **incremental parsing** to force the generation of **valid outputs**.
- GCD is a **unified framework** for structured NLP tasks, leveraging the strengths of existing language models without the need for costly finetuning.

Method

- GCD **constrains** language model outputs at decoding time based on a formal grammar.
- GCD employs an **incremental parser** as a completion engine, guiding the model to produce only **grammatically consistent results**.

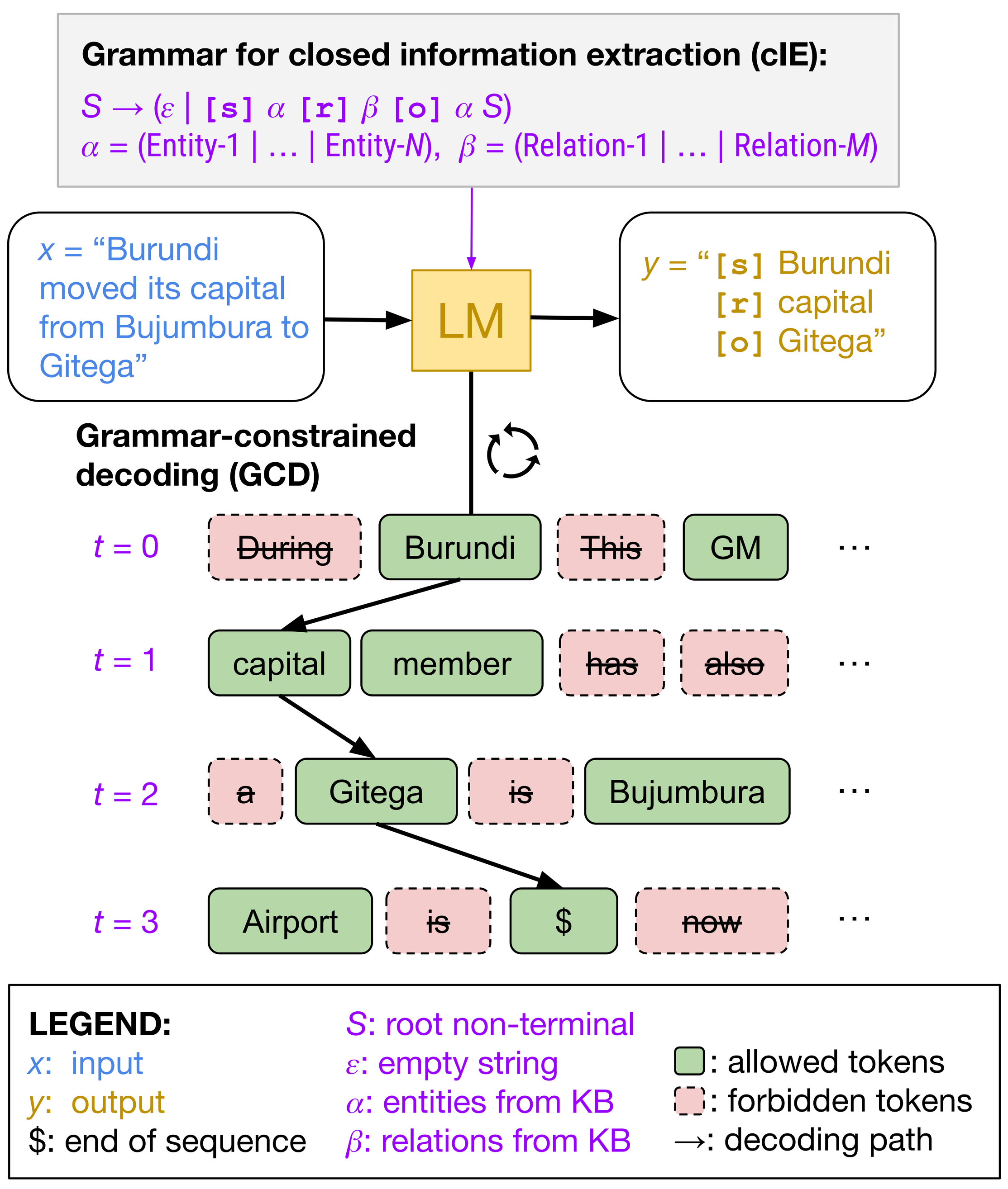

Fig. 1: Grammar-constrained decoding (GCD) applied to IE task

- The goal is to extract a list y of subject–relation–object triplets from the input text x.
- Subjects and objects are constrained to be **Wikidata entities**, e.g. *Ludwig van Beethoven* is a valid subject, but *Ludwig Beethoven* is not.

Experiments

Focus on three NLP tasks:
Closed Information Extraction (cIE): Extracting facts from text using a predefined set of entities and relations.
Entity Disambiguation (ED): Identifying specific entities from a knowledge base mentioned in the text.
Constituency Parsing (CP): Parsing sentences into constituency trees, capturing their syntactic structure.
Each task presents unique challenges for language models, especially in few-shot settings. The experimental setup aims to demonstrate the effectiveness of GCD in improving language model performance across these tasks.

Grammar Gallery

(1) Closed information extraction: see Fig. 1
(2)* Entity disambiguation: $S \rightarrow \ell\ m\ [\alpha]\ r$, where ℓ is left context of mention m, r is right context, and α is disjunction of candidate entities for mention m
(3)* Constituency parsing: $S \rightarrow B_{0,0}$; $B_{i,j} \rightarrow [\alpha\ (B_{i,j+1} \mid C_{i,j+1})$; $C_{i,j} \rightarrow x_i\ (C_{i+1,j} \mid E_{i+1,j})$; $C_{n,j} \rightarrow E_{n,j}$; $E_{i,j+1} \rightarrow\ ]\ (E_{i,j} \mid B_{i,j})$; $E_{n,j+1} \rightarrow\ ]\ E_{n,j}$; $E_{n,0} \rightarrow \varepsilon$, where $\alpha = (S|NP|VP|\ldots)$
(4)* Coreference resolution: $S_i \rightarrow x_i\ [(x_1 \mid \ldots \mid x_n \mid \perp)]\ S_{i+1}$; $S_n \rightarrow \varepsilon$, where $\perp$ means "no referent"
(5)* Part-of-speech tagging: $S_i \rightarrow x_i\ [(NOUN \mid VERB \mid ADJ \mid \ldots)]\ S_{i+1}$; $S_n \rightarrow \varepsilon$
(6)* Dependency parsing: $S_i \rightarrow x_i\ [(ROOT \mid NSUBJ \mid DOBJ \mid \ldots)\ (x_1 \mid \ldots \mid x_n \mid \perp)]\ S_{i+1}$; $S_n \rightarrow \varepsilon$, where $\perp$ means "no head"
(7)* Word sense disambiguation: $S_i \rightarrow x_i\ [\alpha_i]\ S_{i+1}$; $S_n \rightarrow \varepsilon$, where α_i is the disjunction of all WordNet glosses of word x_i
(8)* Phrase chunking: $S \rightarrow B_0$; $B_i \rightarrow [C_i$; $B_n \rightarrow \varepsilon$; $C_i \rightarrow x_i\ (C_{i+1} \mid \alpha]\ B_{i+1})$; $C_n \rightarrow \alpha]$, where $\alpha = (NP \mid VP \mid PP \mid \ldots)$
(9)* Semantic role labeling: Same as phrase chunking, but with $\alpha = (TARGET \mid ARG0 \mid ARG1 \mid \ldots)$
(10)* Entity linking: Same as phrase chunking, but with α the disjunction of all KB entity names (or $\perp$ for "no entity")
(11)* CCG parsing: Same as constituency parsing, but with syntactic types (e.g., (S\NP)/NP)) instead of constituent labels. Extra constraints ensure that nodes have at most two children and that syntactic types combine correctly.
(12)* Question answering: $S \rightarrow [q]\ [A]$; $A \rightarrow (\varepsilon \mid \alpha\ A)$, where q is the question and α the disjunction of all vocabulary words
(13)* Extractive summarization: $S \rightarrow (\varepsilon \mid [\alpha]\ S)$, where α is the disjunction of all sentences from input x
(14)* Semantic parsing with λ-calculus: A logical form is a rooted tree, generated by a context-free grammar

Fig. 2: Formal grammars for 14 structured NLP tasks, highlighting the general applicability of GCD.

Results on IE

Method	Precision	Recall	F1
Weakly supervised			
GenIE T5-base	**49.6** ± 0.3	26.8 ± 0.2	34.8 ± 0.2
Few-shot unconstrained			
LLaMA-7B	10.2± 0.5	14.3± 0.7	11.9± 0.5
LLaMA-13B	10.3± 0.6	17.0± 0.9	12.9± 0.6
LLaMA-33B	14.1± 1.0	23.1± 1.4	17.5± 1.0
Vicuna-7B	12.5± 0.2	16.7± 0.1	14.3± 0.2
Vicuna-13B	13.4± 0.2	15.2± 0.2	14.4± 0.2
Few-shot constrained			
LLaMA-7B	27.9± 0.6	20.2± 0.5	23.5± 0.5
LLaMA-13B	36.2± 0.7	26.5± 0.5	30.6± 0.5
LLaMA-33B	39.3± 0.9	**33.2**± 0.8	**36.0** ± 0.7
Vicuna-7B	25.4± 0.5	15.8± 0.3	19.5± 0.3
Vicuna-13B	38.7± 1.0	19.8± 0.8	26.1± 0.8

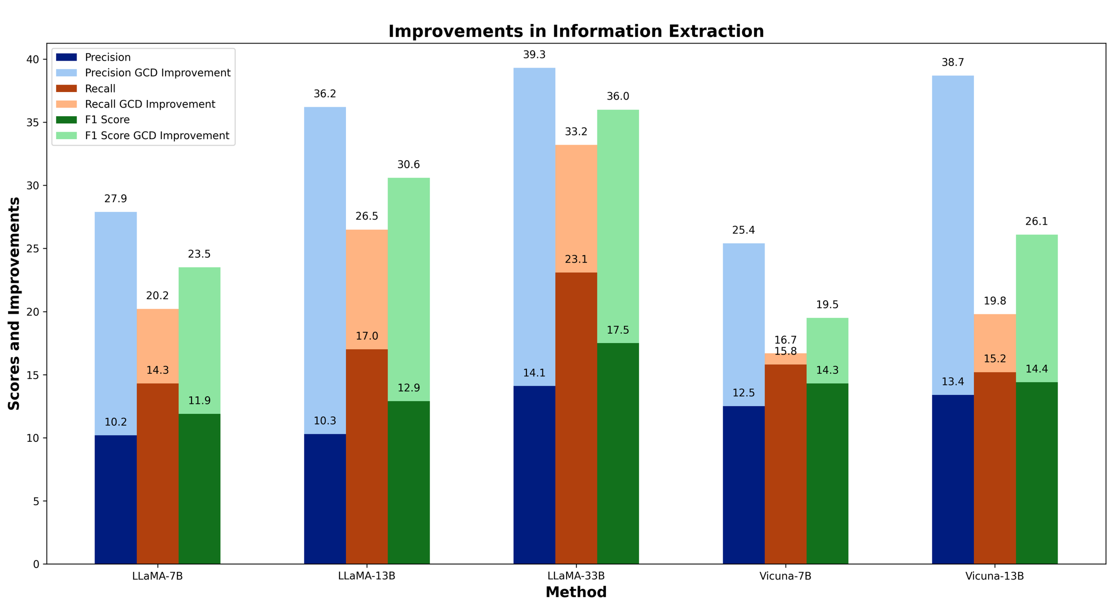

Fig. 3: Grammar-constrained decoding (GCD) applied to IE task

Results on CP

Method	F1	Validity
Few-shot unconstrained		
LLaMA-7B	28.1	54.3
LLaMA-13B	42.8	69.4
LLaMA-33B	42.9	64.2
Few-shot constrained (IDG)		
LLaMA-7B	45.8	100.0
LLaMA-13B	53.4	100.0
LLaMA-33B	54.6	100.0

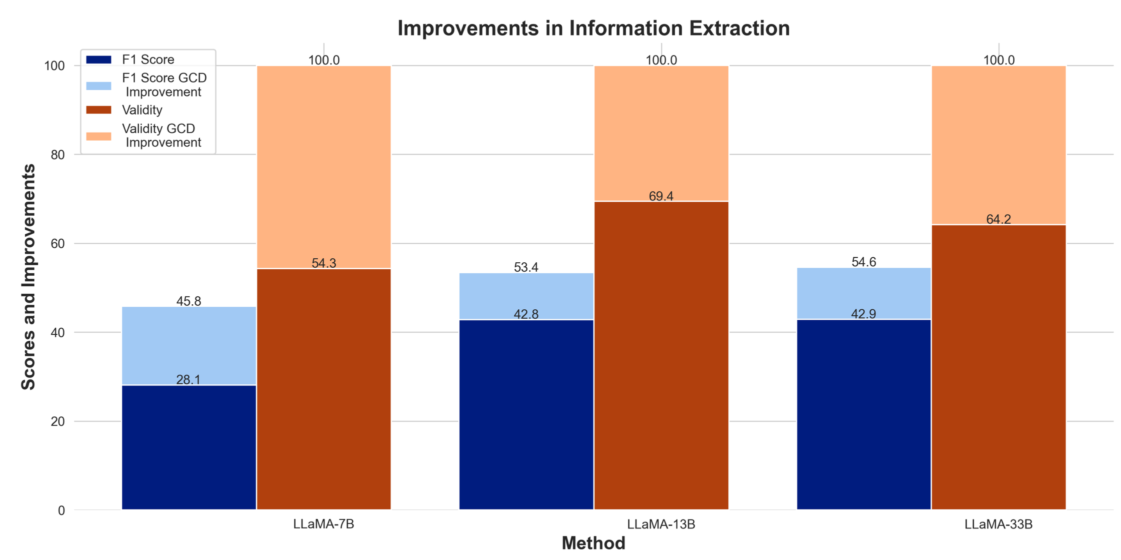

Fig. 4: Grammar-constrained decoding (GCD) applied to CP task

Conclusion

We found that:
- GCD **significantly improves** the performance of LLMs on structured output generation tasks.
- GCD is more effective with larger LLMs. When possible, use the **largest available LLM**.
- Grammars should be as restrictive as possible. Consider using **input-dependent grammars**.
- While GCD is broadly applicable to many tasks, it is not a silver bullet. Tasks that require syntactic understanding of the input (e.g., constituency parsing) are **less suitable** for GCD.

CFG is coming to HuggingFace!

Check out our **CFG** library for **Hugging-Face Transformers** at `https://github.com/` **epfl-dlab** `/transformers-CFG`. Use it to generate **json**, **chess moves**, or even **C code**!

QR Code

paper

code