CHASTE: Constrained Holistic Annotator System with onTological Entities

Task entry of the Graz University of Technology, for the BioASQ Task GutBrainIE on Gut-Brain Interplay Information Extraction at CLEF 2026 in the team ToGS

Benedikt Kantz[1,*], Peter Waldert[1], Stefan Lengauer[1] and Tobias Schreck[1]

[1]Graz University of Technology, Rechbauerstrasse 12, Graz, Austria

Abstract

Entity and Relationship extraction remain key challenges for information extraction in creating knowledge bases from textual resources. This challenge is especially pronounced in domain-specific applications, where systems need to be fine-tuned and tailored to specific needs. This paper, therefore, aims to enhance zero-shot or simple training approaches for knowledge extraction within the GutbrainIE Task 2026 in the Conference and Labs of the Evaluation Forum (CLEF). Hence, we propose a generative grammar-based constrained generation method informed by ontological and natural-language preprocessing, Constrained Holistic Annotator System with onTological Entities (CHASTE), based on Language Models (LMs). The extraction system is evaluated in both fine-tuned and zero-shot settings for entity and relation extraction subtasks. Furthermore, a modified beam search enables the retrieval of joint probabilities over a sequence of tokens, theoretically enhancing grammar-constrained output through a holistic probabilistic generative approach. The results of this evaluation are compared, both within our submission, including a naive simple baseline, and to the official baseline results. A merged run that intersects our and Graphwise teams' results extends these results by leveraging our precise pipeline and securing a top-precision spot in subtask 6.1.2 (Named Entity Recognition and Disambiguation (NERD)). These comparisons show that, while the approach is underperforming, it offers potential for low-training pipelines with easy adaptation, requiring no fine-tuning of a large LM, only our specialized generative pipeline.

Keywords

Natural Language Procession, Named Entity Recognition, Named Entity Disambiguation, Entity Linking, Beam Search, Constrained Natural Language Generation

1. Introduction

Knowledge extraction from natural language texts, especially in domain-specific settings, has been under active research for many years [1]. The core application of this extraction process is the automatic creation of Knowledge Graphs (KGs) within various domains, ranging from material science, fact checking, and medical knowledge for downstream processing and querying [2, 3]. Prior approaches have relied on rule-based or statistical techniques to reliably extract information [3]. Recent advances in language-modeling, among them embedding-based processing and attention mechanics, have spurred new directions in KG construction. With the rising power of generative Language Models (LMs), systems can also directly produce the required information structures [4]. The output of these LMs, however, is, in the default setting, unconstrained and open-ended, leading to invalid structures at best or data hallucination at worst. Existing systems in the Information Extraction (IE) field have, therefore, started to constrain their output to a predefined grammar that adheres to a required schema of entities and their types [5, 6]. These constrained approaches, however, suffer from the "lock-in" once a single token has been predicted, i.e., once a class has been predicted, a structure further down in the generation might not be *probabilistically* advantageous anymore. *Beam search* alleviates this issue by searching for and pruning possible paths

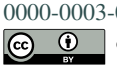
CLEF 2025 Working Notes, September 21–24, 2026, Jena, Germany

*Corresponding author.

benedikt.kantz@tugraz.at (B. Kantz); pwaldert@tugraz.at (P. Waldert); s.lengauer@tugraz.at (S. Lengauer); tobias.schreck@tugraz.at (T. Schreck)

https://dakantz.at/ (B. Kantz); https://github.com/MrP01 (P. Waldert); https://stefan-lengauer.at/ (S. Lengauer)

0000-0003-3294-8421 (B. Kantz); 0009-0004-8459-7381 (P. Waldert); 0000-0001-5136-4320 (S. Lengauer); 0000-0003-0778-8665 (T. Schreck)



for choices, and has been in use for decades in language modelling [7], and is in use within various Machine Learning (ML) systems [8]. There have, however, been no prior integrations of constrained beam search for IE to our knowledge. This paper therefore introduces Constrained Holistic Annotator System with onTological Entities (CHASTE), a novel integration of beam search into generative constrained IE approaches.

Our IE pipeline can handle both Named Entity Recognition (NER) and Relation Extraction (RE) applications through a flexible structural grammar. Both extraction processes follow a similar structure, with

(a.) splitting the text input into sentences for easier downstream processing,
(b.) detecting the name entities using a fine-tuned model,
(c.) building the generative grammar based on the named entities and ontology,
(d.) performing the generation using beam search, and finally
(e.) parsing the generated text.

We evaluate our approach within the GutBrainIE@CLEF 2026 tasl within the medical domain of gut-brain interplay. This evaluation setting provides a strong competitive field, complete with a baseline and hidden test set. The ground truth is extracted from PubMed abstracts[1], which were manually annotated by both medical experts and students, and using a distantly supervised approach, i.e., relations and entities are extracted using an automated system and checked for errors. All entities, additionally, have an URI attached to them, providing an additional challenge for the disambiguation of entities [9].

The task will be introduced in Section 2, and related work on our CHASTE will be presented in Section 3. The complete pipeline is introduced in Section 4, followed by the naive baseline in Section 5. The parameters and results are shown in Sections 6 and 7, respectively. The implications are finally discussed in Section 8. Complete tables extending our results are shown in the supplemental material in our repository on https://github.com/Dakantz/CHASTE.

2. GutBrainIE Task

The GutBrainIE task is hosted as the second edition [9] within the BioASQ laboratory [10] at Conference and Labs of the Evaluation Forum (CLEF) 2026, promoting the advances in IE within the medical subdomain of gut and brain interplay. The task is based on 5081 PubMed abstracts annotated by groups with differing expertise in the subject, ranging from expert-curated annotations in the "gold" collection, followed by "silver" annotations by trained students, and, finally, a large "bronze" collection containing distantly supervised annotated data. These annotations include the entities found in the titles and abstract text, complete with their types and URIs for disambiguation, as well as the relations between them. They are limited to a set of thirteen distinct labels, linked by 25 specific predicate types. The entities shall be extracted within the first subtask, where NER and Named Entity Recognition and Disambiguation (NERD) are evaluated separately; while the RE is covered in the second subtask, again with split evaluation for direct extraction and concept disambiguation over the relations. The results are evaluated by taking the precision P, recall R, and F_1-score for either the macro (i.e. the metric averaged over all types) or micro (i.e. the metric aggregated over all types) averages, while the $F_{1,\text{micro}}$ is used for ranking the participant's results. The articles are split into a training, development (validation), and test set, with 4921, 80, and 80 articles, respectively.

3. Related Works

Existing approaches already incorporate individual aspects of the combination of IE by generative approaches, especially constrained ones, and beam search.

IE from text has seen multiple generations of approaches, beginning with rule-based extraction [1], over ever-more complex embedding-based systems, up to modern generative approaches using transformers [3,

[1] https://pubmed.ncbi.nlm.nih.gov/

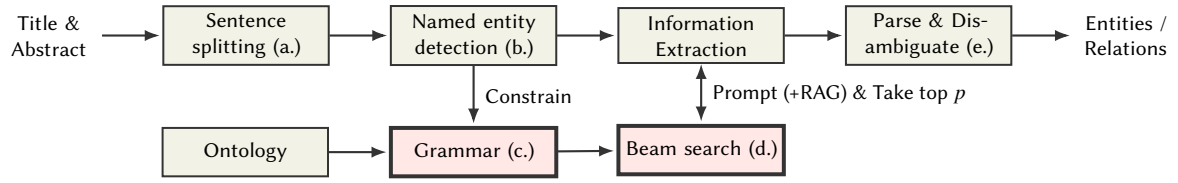


Figure 1: Our IE pipeline. We first extract the sentences (a.) from the title and abstract, then apply a named entity detection system (b.), and finally apply our grammar (c.) with the constrained beam search (d.). The output is then parsed (e.). Our core contributions are highlighted.

5], which model the next token $t \in T$ predictions over a probability p_t, with T being the complete token set. Generative approaches have also already explored constrained generation to limit the output of the LMs to specific schemas. Specifically, [11] incorporated constrained decoding, combined with agentic systems, to perform IE over complete papers with visual information. A prior submission in the GutBrainIE task has also explored constrained generation for IE, but focused only on RE and used complex JSON schemas to generate outputs, resulting in comparatively weak performance [6]. Other systems in query generation have already made use of constrained generation [12], and it is a highly contested field within computer-human interaction [13].

Beam search offers an interesting possible extension avenue for this constrained generation, as the predictions of a generative model can be sampled further down and traced back to achieve an advantageous overall probability, instead of just inspecting the direct next hit [7]. It has already been investigated for Long short-term memory (LSTM) models for constrained decoding [8] and also applied within machine translations [14].

Our work, therefore, aims to fill this gap in the literature by experimenting with both branches and evaluating the potential of constrained decoding combined with beam search for improved generative information retrieval from text in the medical domain.

4. Annotation Process

CHASTE consists of an extraction pipeline shown in Figure 1. Each step is outlined in the following section, beginning with data preprocessing.

Preprocessing (a.) This step primarily serves to keep the length of the downstream inputs short, as the last generative submission indicated difficulty generating enough outputs to be viable [6]. We therefore split the initial titles and abstracts of the PubMed articles in the task into sentences and additionally preprocessed them to ensure their efficiency and effectiveness. The locations of the entities, specifically their offsets, are recalculated to remain in the correct spaces for the prediction, and are recalculated after the predictions have been made to join the sentences back into the abstracts.

Named Entity Detection (b.) Each sentence is then evaluated within our IE pipeline, first extracting candidate locations for named entities. These are needed for a more fine-grained grammar approach, using the accuracy of these systems to inform our grammar within the downstream process.

We use two different approaches within this pipeline stage: first, we employ an off-the-shelf solution, namely spaCy [15], using its transformer-based NER for Named Entity Detection (NED), which serves as the baseline for extracting named entities from sentences. We additionally fine-tune both a `DistilBERT` [16] and `PubMedBERT-MNLI-MedNLI` [17] for token-level annotation for domain adaptation of the extraction process. This is crucial, since the next pipeline steps heavily depend on accurate entity spans for the correct grammar generation. We choose the PubMed-finetuned variant for the downstream subtasks, as it outperformed the base `DistilBERT`.

Listing 1 The corresponding entity grammar of the sentence "Orthopedic Surgery Causes Gut Microbiome Dysbiosis and Intestinal Barrier Dysfunction in Prodromal Alzheimer Disease Patients: A Prospective Observational Cohort Study.".

```
root ::= ent-list
entity ::= entity-type" ("entity-str")"
entity-type ::= "Anatomical Location"|"Animal"|"Biomedical
↪  Technique"|"Bacteria"|"Chemical"|"Dietary
↪  Supplement"|"DDF"|"Drug"|"Food"|"Gene"|"Human"|"Microbiome"|"Statistical Technique"
entity-str ::= "Orthopedic Surgery"|"Gut Microbiome Dysbiosis"|"Intestinal Barrier
↪  Dysfunction"|"Prodromal Alzheimer Disease Patients"|"Prospective Observational Cohort
↪  Study"
ent-list ::= entity ("\\n" entity)*
```

Grammar Generation (c.) The generated grammar for the generative IE is informed by two aspects: the ontology and the previously extracted named entities. This, however, requires a distinct grammar for each generated sentence to ensure the model returns only valid structures. We employ the GGML Backus-Naur Form (GBNF) and build it for every sentence passed through the pipeline, separately for the entity and relation subtasks. A grammar for the first sentence in the first article of the dev set could therefore be formulated as shown in Listing 1. The grammar for the relations is built similarly, with just the additional subject, predicate, and object structure for each predicted instance. We refer to the provided repository for further details. The grammar thereby constrains the model through both the ontology and a strong natural language prior.

Constrained Generation with Beam Search (d.) Finally, once the grammar is built, the information extraction can commence. We use a combination of strategies for this generative approach, comparing a range of possible options. We employ Retrieval Augmented Generation (RAG), Low-Rank Adapation (LoRA) [18] fine-tuning, different models, and beam search using two variations.

First, RAG is applied to provide the model with prior examples in a highly cost-effective way, since it does not require any prior fine-tuning. We use the `NeuML/pubmedbert-base-embeddings` model, based on `sentence-bert` [19], to retrieve k semantically similar sentences from the training set and present these in the same format as required by grammar to the model in a multi-turn chat setting, before starting the predictions of either entities or relations. We removed the reweighing by annotator class (i.e., gold, silver, and bronze), as last year's entries did not achieve improvements when employing it [6].

Next, we fine-tune LMs separately for both entity and relation extraction. The fine-tuning uses the LoRA approach [18] to remain resource-efficient, while still improving the model output. The models are fine-tuned by presenting each article as a multi-turn example across all sentences within it, mimicking the k nearest-neighbor sentences at inference time.

All these approaches are applied to two models within the same family, namely Hermes 3 [20]. These are less recent than other models but are still quite efficient in terms of parameter count, positioning them well for our use case. They are also fine-tuned for such structured output subtasks, making them suitable for our application. These fine-tuned models, along with their base models, are used to generate the tokens based on the grammar of (c.).

Additionally, the generation of tokens can be adjusted using constrained beam search [7, 14], if enabled. The beam search utilizes the grammar and the LMs, fine-tuned or not, to generate complete passages in a holistic manner. The algorithm of Algorithm 1 describes how we combine the tree search with the constraints imposed by the grammar, and the probabilities provided by the LM. The recursive search expands each step by a factor f determined by a predefined list, and filters out all non-matching tokens t based on the grammar derived from the named entities and ontology, taking only the top f options at a maximum. Once the search reaches the maximum depth d_m or an end token, we stop the search, aggregate all probabilities. We use the negative log-likelihood $-\log p_t$ in our implementation to improve

Algorithm 1 Our beam search algorithm constrained to the grammar for both search types, (i.) and (ii.). The search is configured by having a list of expansion factors, possible end token $t_{\text{end,LLM}}$ or maximum depth $d_{\max}$. A LLM with a grammar is required, with a prompt set at the start. The prompt might already contain prior generated tokens t.

Require: expansion_factor $= [f_1, \dots f_j], d_{\max} \in \mathbb{N}^+, t_{\text{end,LLM}}$, LLM, grammar, prompt
  procedure search_node(node, d)
    node.children = [ ]
    for $t, p_t \in$ LLM.predict(node.tokens) **do** ▷ *Get the probability p_t over all tokens t from the LLM*
      if grammar.allows(nd.text + t) **then** ▷ *Filter out tokens based on grammar*
        node.children.append(node.text + t, p_t)
    if $d < j$ **then**
      $f \leftarrow$ expansion_factor$[d]$
    else
      $f \leftarrow f_j$
    for c$\in$ node.children.top(f) **do** ▷ *Get top f tokens.*
      if not is_end(c) **then**
        if $t_{\text{end,LLM}}$ **is defined** | $d < d_{\max}$ **then**
          ▷ *Go only deeper if within maximum depth $d_{\max}$ or if we have a specific end token $t_{end,LLM}$ defined.* ◁
          search_node(c, $d+1$)
  node$_r$.text$\leftarrow$ prompt ▷ *The root node with the prompt.*
  search_node(node$_r$, 0) ▷ *Search nodes and build hierarchy.*
  initialize node.$p_d \leftarrow$ node.p_t $\forall$node $\in$ node$_r$.descendants
  procedure aggregate_probabilites(node) ▷ *Aggregate all probabilities.*
    node.p_d = node.$p_d \cdot$ node.parent.p_d **if** node **not** root
    for $c \in$ node.children **do**
      aggregate_probabilites(c)
  aggregate_probabilites(nd_r)
  function find_max(node) ▷ *Find the best* leaf *probability.*
    return arg max$_{c\in\text{node.children}}$ find_max(c).p_d
  $n_{\text{leaf}} \leftarrow$find_max($nd_r$)

numerical stability, as

$$\arg \max_{c \in \text{node.children}} \text{find\_max}\,(c)\, c.p_d$$

amounts to the same result as

$$\arg \max_{c \in \text{node.children}} -\log \text{find\_max}\,(c)\, c.p_d$$

with the aggregation becoming

$$-\log \text{node}.p_d \leftarrow -\log \text{node.parent}.p_d - \log p_d,$$

due to the monotonic property and product rule of the logarithm. We therefore find the leaf with the best probability even with the negative log-likelihood. The unique path from the root node nd_r to the leaf node nd_{leaf} then holds the, by the model predicted, most probable *combination* of nodes over the prompt, grammar, and evaluated depths. Furthermore, two distinct approaches emerge by varying the input parameters:

(i.) a complete approach, always expanding the search by a factor f of tokens, requiring exponentially more tokens with d – requiring a limited, shallow search, or

(ii.) an initially broad search, which is then limited to only the top node until the end token is possible within the grammar.

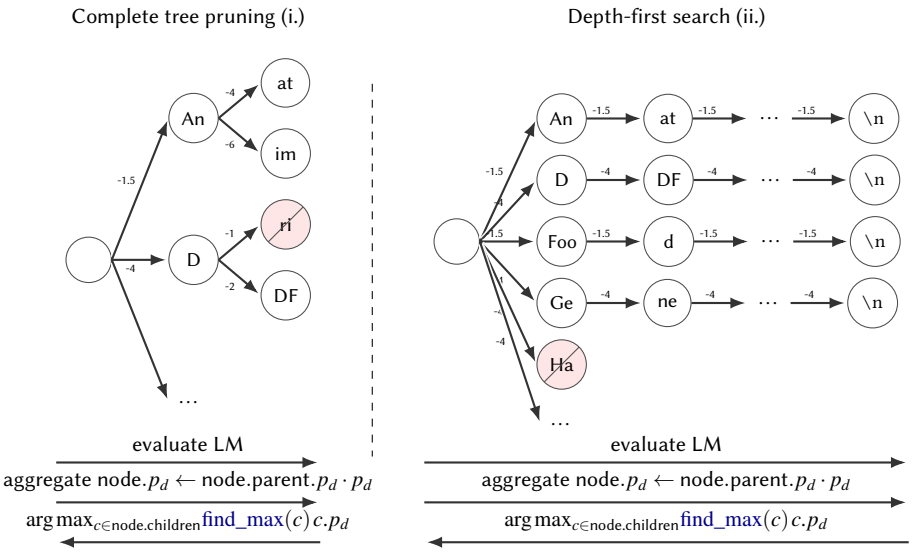


Figure 2: Our two beam search approaches. The left shows how we prune a classical evaluated tree (i.), while the right illustrates how our complete evaluation to the end of one predicted entity searches for the best match (ii.). The numbers on the nodes represent the negative log-likelihood $\log p_t$ for the respective tokens.

The approaches are illustrated in Figure 2, showing how these two divergent approaches generate tokens over the grammar and can result in vastly different predictive power and speed. It is also clear from this illustration that *each* generated token in (i.) requires $\prod_i d_i$ tokens, whereas the depth-first search resolves to a complete entity directly, delimited by the end token, therefore only requiring the estimated initial overhead of d_1 tokens per final token, assuming the generated entities are of the same length. The second approach also allows direct adaptation of the complete sequence until the end token, unlike the shallow search, where more tokens *can* be adapted but might not be advantageous.

Parsing and Disambiguation (e.) Parsing the output requires comparatively fewer resources, as we can just use regular expressions on it, since we already ensure the correct format with the grammar. The parsed information is then localized based on the named entities extracted in the previous step and enhanced by the Unique Resource Identifiers (URIs). Our URIs prediction system is quite simple: as we already know all correct matching entity texts, we can just do a direct match with this collection of strings, and find the correct one. To enhance the possibility of out-of-collection, we employ Term Frequency - Inverse Document Frequency (TF-IDF) to find such close matches.

5. Naive Baseline

A naive baseline expands our range of tests by introducing a purely lookup-based system for detecting and classifying entities. Its purpose is two-fold; first, we can compare to a more sanity-check-based driven baseline with easy debug-ability and a fast runtime, and, second, we establish a secondary baseline that can be of use for other participants in next year's task that does not require elaborately trained systems, but a simple and straightforward system that allows easy incorporation and comparisons. A third use case emerges through development, as we use these grounded entities to enrich the grammar for the IE in from Section 4.

The baseline explores a very naive approach to tagging words, as outlined in Algorithm 2. The algorithm searches for exact matches of each entity in a given sentence, then annotates the sentence as-is. We adhere to the annotation rules by stripping entity texts and omitting short strings. Once the matches within a sentence are found, they can optionally be filtered by checking for possible overlaps across the found entities.

Algorithm 2 The naive NER algorithm for annotating entities based on the training set. We filter optionally on entities that might have (longer) overlaps with others.

Require: sentence s, entities $E = \{(c_1, s_1, u_1), \ldots\}$ with concepts c, span s, and identifier u, enable filter flag.

```
E_f ← [ ]
for (c_e, s_e, u_e) ∈ E do
    M ← find_matches(s_e, s)
    for m ∈ M do
        E_f.append((c_e, m, u_e))
if enabled filter then
    E_{f,filtered} ← {(c_e, s_e, u_e) ∈ E_f | not any_overlap(s_e, E_f)}
```

Table 1
The parameters used to train the NED model `DistilBERT` [16]. The `DistilBERT` [16] uses the same parameters, except for the number of epochs (10), for its fine-tuning run.

Parameter	Value
learning rate	10^{-4}
batch size	32
epochs	16
weight decay	0

Table 2
The parameters used to train the Hermes models [20, 18].

Parameter	Value
learning rate	10^{-3}
batch size	3
epochs	3
weight decay	10^{-2}

6. Evaluation Setup

Applying the algorithms of Sections 4 and 5 for the GutbrainIE task [21] requires, initially, setting parameters for the training of the NED first, and configuring the generative model training and inference. Once they are set up, the models can be trained. These settings and training results are shown in Sections 6.1 and 6.2, respectively.

6.1. Training and Model Parameters

The selection process for all pipeline stages is based limited experiments, library defaults, and hardware and resource constraints, with the details of the final used parameters shown in the following paragraphs.

Named Entity Detection The NED uses the model discussed above, the `PubMedBERT-MNLI-MedNLI` [17]. This model is fine-tuned to tag entities as candidates for grammar-based detection, using the parameters in Table 1.

Generative Models Next in the pipeline, the generative models constrained by the grammar employ the Hermes 3.1 8B and 3.2 3B models, tuned to generate structured outputs from the full training sets, using the parameters in Table 2, separately for entities and relations. The trained models are used for inference on all non-naive evaluation runs using the LoRA flag, with the parameters in Table 3.

Table 3
The parameters used to run inference on the finetuned models.

Parameter	Value
temperature	10^{-1}
context size	8196
top-k	5
maximum tokens t	512

Table 4
Evaluation of NED annotators on the Development Set.

Type	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
PubMedBERT	**0.75**	0.69	**0.72**	**0.75**	0.69	**0.72**
DistilBERT	0.68	0.64	0.66	0.68	0.64	0.66
Spacy	0.34	**0.80**	0.48	0.34	**0.80**	0.48

Table 5
Top 10 Development Set Results for NER (Subtask 6.1.1).

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	×	✓	×	×	✓	×	0.49	0.76	0.58	0.49	0.78	0.60
				✓	✓	×	0.48	0.78	0.59	0.47	0.79	0.59
			✓	✓	✓	×	0.50	0.78	0.60	0.48	0.80	0.60
	End	✓	×	×	✓	×	**0.51**	0.76	0.60	**0.52**	0.78	**0.63**
3.2 3B	×	✓	×	×	✓	×	0.49	0.76	0.58	0.49	0.78	0.60
				✓	✓	×	0.50	0.78	0.60	0.47	0.79	0.59
			✓	×	✓	×	0.48	0.76	0.58	0.48	0.78	0.59
	End	✓	×	✓	✓	×	0.50	0.74	0.59	0.50	0.77	0.61
			✓	✓	✓	×	0.50	0.73	0.58	0.50	0.76	0.60
Naive	×	×	×	×	✓	×	0.49	**0.80**	**0.60**	0.51	**0.81**	0.62

6.2. Named Entity Detection

The first result we investigate is the raw NED, which is no longer part of this year's task. Nonetheless, investigating these results provides solid ground for our reasoning in the evaluation campaign of Section 7. We observe that fine-tuning a BERT model, in this case `DistilBERT` [16], already yields substantial improvements over our Spacy baseline, as shown in Table 4. Using a domain-specific PubMedBERT (`PubMedBERT-MNLI-MedNLI` [17]) further improves these scores, especially F_1 and precision. The recall is still best for the Spacy detection mechanism, and general performance could be improved to catch up to the best models of the task – since this detection mechanism constrains the downstream pipeline, and, in turn, places a hard upper limit on the achievable performance.

7. Evaluation Campaign Resultus

Integrating these methodologies, especially the beam search and naive baseline, and the setups from above, we can run our evaluation campaigns over a breadth of combinations. The runs are enriched by merges with the Graphwise team, using their top results and taking both the intersection ($\cap$) and union ($\cup$) over the extracted information. This section shows only the top 10 results for each task; the appendix in our source repository provides the complete results. We also include the organizers' baseline as a reference for our test results.

Table 6
Top 10 Development Set Results for NED (Subtask 6.1.2).

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	×	✓	×	×	✓	×	0.42	0.66	0.50	0.41	0.66	0.51
				✓	✓	×	0.42	0.67	0.51	0.40	0.66	0.50
			✓	✓	✓	×	0.43	0.67	0.52	0.40	0.67	0.50
	End	✓	×	×	✓	×	0.44	0.65	0.51	0.43	0.64	0.52
3.2 3B	×	✓	×	×	✓	×	0.43	0.67	0.51	0.42	0.67	0.51
			✓	✓	✓	×	0.43	0.67	0.51	0.40	0.66	0.50
	End	✓	×	✓	✓	×	0.44	0.64	0.51	0.42	0.64	0.50
			✓	✓	✓	×	0.43	0.64	0.51	0.42	0.64	0.51
Naive	×	×	×	×	✓	×	0.43	**0.69**	0.52	0.42	**0.67**	0.52
						✓	**0.50**	0.62	**0.54**	**0.52**	0.59	**0.56**

Table 7
Development Set Improvements for NER (Subtask 6.1.1) when applying beam search.

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	End	✓	×	×	✓	×	+0.02	-0.01	+0.02	+0.03	-0.00	+0.02
				✓	✓	×	+0.01	-0.43	-0.19	+0.04	-0.45	-0.18
			✓	×	×	×	+0.00	+0.00	+0.00	+0.00	0.00	+0.00
				✓	✓	×	+0.01	-0.09	-0.02	+0.03	-0.10	-0.01
3.2 3B	End	✓	×	✓	✓	×	+0.01	-0.04	-0.01	+0.03	-0.02	+0.02
			✓	×	×	×	+0.00	+0.00	+0.00	-0.00	-0.00	-0.00
					✓	×	+0.04	-0.12	-0.02	+0.04	-0.15	-0.03
				✓	✓	×	+0.01	-0.03	-0.00	+0.03	-0.03	+0.01

Table 8
Top 10 Development Set Results for RE (Subtask 6.2.1).

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	×	×	×	✓	×	×	**0.07**	0.09	0.04	**0.08**	**0.11**	**0.10**
					✓	×	0.07	**0.09**	0.04	0.08	0.11	0.09
			✓	×	×	×	0.05	0.07	**0.05**	0.06	0.09	0.07
					✓	×	0.05	0.07	**0.05**	0.06	0.09	0.07
				✓	×	×	0.04	0.06	0.03	0.07	0.08	0.07
					✓	×	0.04	0.06	0.03	0.07	0.08	0.07
3.2 3B	×	×	×	✓	×	×	0.04	0.04	0.03	0.08	0.11	0.09
					✓	×	0.03	0.04	0.03	0.08	0.10	0.09
			✓	✓	×	×	0.04	0.05	0.03	0.07	0.09	0.08
					✓	×	0.04	0.05	0.03	0.07	0.09	0.08

7.1. Development Evaluation Results

Beginning with the development ("dev") set validation runs, we take almost the complete combination of ⟨Beam Search, NED fine-tune, RAG, LoRA, Naive enrichment⟩ flags and add our baseline for NER with and without filter. The results are partitioned by our (ToGS) results and merged with Graphwise, where we only consider the top results from our own evaluation campaign.

ToGS results Beginning with our scores for the NER in Table 5 and NERD inTable 6, we observe, compared to our more advanced pipeline, already reasonable results for the naive IE pipeline; proving

Table 9
Top 10 Development Set Results for Mention-level Relation Extraction (M-RE) (Subtask 6.2.2).

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	×	×	×	✓	×	×	0.06	0.09	0.05	**0.07**	**0.11**	**0.09**
					✓	×	0.06	0.09	0.05	0.07	0.10	0.08
			✓	×	×	×	**0.07**	**0.11**	**0.07**	0.06	0.10	0.07
					✓	×	**0.07**	**0.11**	**0.07**	0.06	0.10	0.07
3.2 3B	×	×	×	✓	×	×	0.06	0.05	0.04	0.07	0.10	0.08
					✓	×	0.06	0.05	0.04	0.07	0.10	0.08
			✓	×	×	×	0.05	0.07	0.05	0.05	0.09	0.06
					✓	×	0.05	0.07	0.05	0.05	0.09	0.06
				✓	×	×	0.05	0.04	0.03	0.05	0.07	0.06
					✓	×	0.05	0.04	0.03	0.05	0.07	0.06

Table 10
Development Set Improvements for M-RE (Subtask 6.2.2) when applying beam search.

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
3.1 8B	End	✓	×	✓	×	×	-0.00	-0.00	-0.00	-0.00	-0.01	-0.01
					✓	×	+0.00	-0.00	-0.00	+0.00	-0.01	-0.01
			✓	×	✓	×	-0.01	-0.01	-0.01	+0.00	-0.01	-0.01
				✓	×	×	-0.00	-0.00	-0.00	+0.00	-0.01	-0.00
3.2 3B	End	✓	×	✓	×	×	-0.02	-0.03	-0.01	-0.01	-0.02	-0.02

Table 11
Top 10 Merged Development Set Results for NER (Subtask 6.1.1), including the results of the merged approaches with Graphwise.

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
NER-EXP-SUB-EL-freq-gaze	∩	3.1 8B	×	✓	✓	0.89	0.70	0.76	0.90	0.74	0.81
		3.2 3B	×	✓	✓	0.89	0.70	**0.76**	0.90	0.74	**0.81**
			End	✓	✓	**0.90**	0.68	0.75	**0.90**	0.72	0.80
		Naive	×	×	×	0.89	0.70	0.76	0.90	0.74	0.81
	∪	3.2 3B	End	✓	✓	0.52	0.80	0.62	0.55	0.87	0.67
		Naive	×	×	×	0.50	0.81	0.61	0.53	0.88	0.66
NER-EXP-SUB	∪	3.1 8B	×	✓	✓	0.50	0.81	0.61	0.51	0.87	0.64
		3.2 3B	×	✓	✓	0.50	0.80	0.61	0.51	0.87	0.64
			End	✓	✓	0.52	0.81	0.62	0.55	0.87	0.68
		Naive	×	×	×	0.51	**0.82**	0.62	0.53	**0.88**	0.66

that it is, indeed, a good baseline for our system. We note that we were unable to improve the baseline provided by the task [21] for any of our approaches. Our evaluation campaign, nonetheless, reveals interesting patterns, most notably improvements of the beam search approach over approaches without it, as made explicit in Table 7. The precision of the pipeline is enhanced across the board, while recall suffers by a large margin - making it a good contender for generative pipelines seeking to focus on precision over recall. The results using the shallow beam search, however, did not yield any usable results – while completing after exorbitantly more time to the exponential increase in tokens generated per depth step. The best combination of approaches, i.e., the one with NED fine-tunes, no RAG, and, most interestingly, no LoRA, but enrichment by the naive approach, results in increased performance, positioning it at the top of our results. Another interesting aspect is the very strong recall of our naive

Table 12
Top 10 Merged Development Set Results for NED (Subtask 6.1.2), including the results of the merged approaches with Graphwise.

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
NER-EXP-SUB-EL-freq-gaze	∩	3.1 8B	×	✓	✓	0.86	0.67	0.74	0.85	0.70	0.77
		3.2 3B	×	✓	✓	0.86	0.67	**0.74**	0.85	0.70	**0.77**
			End	✓	✓	**0.87**	0.65	0.73	**0.86**	0.69	0.76
		Naive	×	×	×	0.86	0.67	0.74	0.85	0.70	0.77
	∪	3.1 8B	×	✓	✓	0.43	0.69	0.52	0.42	0.72	0.53
		3.2 3B	×	✓	✓	0.44	0.69	0.53	0.42	0.72	0.53
			End	✓	✓	0.45	0.69	0.53	0.46	0.72	0.56
		Naive	×	×	×	0.44	**0.70**	0.53	0.44	**0.73**	0.55
NER-EXP-SUB	∪	3.2 3B	End	✓	✓	0.24	0.37	0.29	0.23	0.37	0.29
		Naive	×	×	×	0.25	0.40	0.30	0.23	0.39	0.29

Table 13
Top 10 Merged Development Set Results for RE (Subtask 6.2.1), including the results of the merged approaches with Graphwise.

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
gpt-5.shot-gold	∩	3.2 3B	×	×	✓	0.17	0.03	0.05	**0.69**	0.09	0.17
	∪	3.2 3B	×	×	✓	**0.24**	**0.59**	**0.31**	0.24	**0.64**	**0.35**

Table 14
Top 10 Merged Development Set Results for M-RE (Subtask 6.2.2), including the results of the merged approaches with Graphwise.

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
gpt-5.shot-gold	∩	3.2 3B	×	×	✓	0.22	0.04	0.06	**0.62**	0.08	0.15
	∪	3.2 3B	×	×	✓	**0.23**	**0.60**	**0.30**	0.21	**0.63**	**0.31**

approach on the development set, positioning it as a strong contender for merging with Graphwise. We additionally note that the best pipeline combination *with* beam search achieves its performance by just fine-tuning the small NED model, requiring no full LoRA of the large tagging model.

Moving on to Subtask 6.2, i.e., RE and M-RE, shown in Tables 8 and 9[2] we observe much worse results compared to the development set results on RE in Table 5 – comparable to the performance of the constrained generation of last year's task [6]. We further observe almost no change in performance when applying beam search to constrained generation in Table 10, suggesting limited efficiency for more complex extraction tasks, such as RE.

Analyzing the progression over the scores when limiting the annotations to the first m predictions and ground truth annotations in Figure 3, i.e., cutting number of annotations to calculate the scores, also reveals interesting properties of the IE pipeline. Our approach seems to generate too many annotations relative to the ground truth, as reflected in this low precision and high recall. This is also evident in the very strong initial rise in precision up to the mean ground-truth annotation counts.

Merged results Merging our results with the strong approach of Graphwise yields a comparatively precise system for NED, achieving up to 0.9 and a high F_1 score of around 0.81, compared to our precision of around 0.5. This is evident by Tables 11 to 14. The results for 6.2.2 reveal similarly good precision.

[2]Runs without retrieved relations and with a score of 0 were omitted for brevity.

Table 15
Top 10 Test Set Results for NER (Subtask 6.1.1)

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
Baseline	-	-	-	-	-	-	**0.71**	**0.75**	**0.73**	**0.78**	**0.82**	**0.80**
3.1 8B	×	×	✓	×	✓	×	0.34	0.45	0.37	0.35	0.58	0.44
		✓	×	×	✓	×	0.31	0.45	0.35	0.34	0.53	0.42
				✓	✓	×	0.31	0.45	0.35	0.33	0.53	0.41
			✓	✓	✓	×	0.33	0.45	0.37	0.33	0.53	0.41
3.2 3B	×	×	×	✓	✓	×	0.32	0.39	0.33	0.34	0.55	0.42
			✓	×	✓	×	0.33	0.44	0.36	0.34	0.57	0.43
				✓	✓	×	0.32	0.39	0.33	0.34	0.55	0.42
		✓	×	×	✓	×	0.30	0.44	0.34	0.34	0.53	0.42
Naive	×	×	×	×	✓	×	0.31	0.46	0.36	0.35	0.54	0.43

Table 16
Top 10 Test Set Results for NED (Subtask 6.1.2)

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
Baseline	-	-	-	-	-	-	**0.38**	**0.40**	**0.39**	**0.43**	**0.45**	**0.44**
3.1 8B	×	✓	×	×	✓	×	0.22	0.31	0.24	0.24	0.38	0.30
				✓	✓	×	0.22	0.31	0.25	0.23	0.37	0.28
			✓	✓	✓	×	0.24	0.32	0.26	0.24	0.38	0.29
3.2 3B	×	×	×	✓	✓	×	0.22	0.26	0.22	0.22	0.36	0.28
		✓	×	×	✓	×	0.21	0.30	0.23	0.24	0.37	0.29
				✓	✓	×	0.23	0.30	0.25	0.23	0.37	0.28
			✓	✓	✓	×	0.23	0.30	0.25	0.23	0.37	0.28
Naive	×	×	×	×	✓	×	0.22	0.32	0.25	0.25	0.38	0.30
						✓	0.25	0.25	0.24	0.31	0.31	0.31

Table 17
Top 10 Test Set Results for RE (Subtask 6.2.1)

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
Baseline	-	-	-	-	-	-	**0.37**	**0.29**	**0.30**	**0.44**	**0.35**	**0.39**
3.1 8B	×	×	×	✓	×	×	0.11	0.08	0.06	0.11	0.12	0.11
					✓	×	0.11	0.08	0.06	0.11	0.12	0.11
			✓	×	×	×	0.09	0.07	0.06	0.07	0.09	0.08
				✓	×	×	0.04	0.05	0.02	0.10	0.10	0.10
					✓	×	0.04	0.05	0.02	0.10	0.10	0.10
3.2 3B	×	×	×	✓	×	×	0.08	0.06	0.05	0.12	0.11	0.11
					✓	×	0.08	0.06	0.05	0.12	0.11	0.11
			✓	✓	×	×	0.04	0.05	0.02	0.11	0.10	0.10
					✓	×	0.04	0.05	0.02	0.11	0.10	0.10

Both results also highlight a tendency for the intersection to promote precision and the union to optimize for recall. All tested top merges use naive enrichment and do not use RAG.

7.2. Test Evaluation Results

Following the dev results, we also provide the test evaluation results filtered for our systems and merges. First, we observe a strong drop in performance for NED across the board, for both our own and the

Table 18
Top 10 Test Set Results for M-RE (Subtask 6.2.2)

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
Baseline	-	-	-	-	-	-	**0.10**	**0.10**	**0.10**	**0.14**	**0.13**	**0.13**
3.1 8B	×	×	×	✓	×	×	0.05	0.03	0.03	0.06	0.07	0.06
					✓	×	0.05	0.03	0.03	0.06	0.07	0.06
			✓	✓	×	×	0.02	0.03	0.01	0.05	0.06	0.06
					✓	×	0.02	0.03	0.01	0.05	0.06	0.06
3.2 3B	×	×	×	✓	×	×	0.03	0.04	0.03	0.06	0.07	0.06
					✓	×	0.03	0.04	0.03	0.06	0.07	0.06
			✓	×	×	×	0.03	0.04	0.03	0.04	0.05	0.04
				✓	×	×	0.02	0.03	0.01	0.05	0.05	0.05
					✓	×	0.02	0.03	0.01	0.05	0.05	0.05

Table 19
Development Set Disambiguation Changes for 6.1 between NER and NED.

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	F_1	$F_{1,micro}$	Relative F_1	Relative $F_{1,micro}$
3.1 8B	×	✓	×	✓	✓	×	-0.08	-0.10	-0.13	-0.16
			✓	✓	✓	×	-0.08	-0.10	-0.13	-0.17
	End	✓	×	×	✓	×	-0.08	-0.11	-0.14	-0.17
3.2 3B	×	✓	×	×	✓	×	-0.07	-0.09	-0.12	-0.15
				✓	✓	×	-0.08	-0.10	-0.14	-0.17
			✓	×	✓	×	-0.08	-0.11	-0.14	-0.18
				✓	✓	×	-0.07	-0.10	-0.12	-0.16
	End	✓	×	✓	✓	×	-0.08	-0.10	-0.13	-0.17
			✓	✓	✓	×	-0.08	-0.10	-0.13	-0.16
Naive	×	×	×	×	✓	×	-0.08	-0.10	-0.13	-0.17

Table 20
Test Set Disambiguation Changes for Subtask 6.1 between NER and NED.

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	F_1	$F_{1,micro}$	Relative F_1	Relative $F_{1,micro}$
Baseline	-	-	-	-	-	-	-0.34	-0.36	-0.46	-0.45
3.1 8B	×	×	✓	×	✓	×	-0.16	-0.17	-0.42	-0.39
		✓	×	×	✓	×	-0.10	-0.12	-0.30	-0.29
				✓	✓	×	-0.11	-0.12	-0.31	-0.30
			✓	✓	✓	×	-0.11	-0.11	-0.29	-0.28
3.2 3B	×	×	✓	×	✓	×	-0.14	-0.16	-0.40	-0.37
		✓	×	×	✓	×	-0.11	-0.13	-0.32	-0.31
				✓	✓	×	-0.12	-0.12	-0.32	-0.30
			✓	✓	✓	×	-0.12	-0.12	-0.32	-0.31
Naive	×	×	×	×	✓	×	-0.11	-0.12	-0.30	-0.29

merged results. The second limitation of our results is that we could not submit any results with beam search enabled due to time constraints, as they require linearly more tokens – better than the shallow beam search, but worse than the base constrained generation. The inclusion of these, however, would most likely have led to only marginal improvements, as already indicated in the development set.

ToGS results Our own results in Tables 15 to 18, first, indicate a strong discrepancy between the test and development set, leading to worse results of our system across the evaluation in NED, and small, but

Table 21
Test Set Disambiguation Changes for Subtask 6.2 between RE and M-RE.

Model	Beams	NED FT	RAG	LoRA	Naive	Filter	F_1	$F_{1,micro}$	Relative F_1	Relative $F_{1,micro}$
Baseline	-	-	-	-	-	-	-0.20	-0.25	-0.68	-0.65
3.1 8B	×	×	×	✓	×	×	-0.04	-0.05	-0.61	-0.43
					✓	×	-0.04	-0.05	-0.61	-0.43
			✓	×	×	×	-0.03	-0.03	-0.46	-0.42
					✓	×	-0.03	-0.03	-0.46	-0.42
3.2 3B	×	×	×	✓	×	×	-0.02	-0.05	-0.46	-0.44
					✓	×	-0.02	-0.05	-0.46	-0.44
			✓	×	×	×	-0.02	-0.02	-0.40	-0.32
					✓	×	-0.02	-0.02	-0.40	-0.32
				✓	×	×	-0.01	-0.05	-0.52	-0.51

Table 22
Top 10 Merged Test Set Results for NER (Subtask 6.1.1)

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
		Baseline	-	-	-	0.71	0.75	**0.73**	0.78	0.82	**0.80**
ENS-EXP-SUB-1	∪	Naive	×	×	×	0.48	0.75	0.57	0.52	0.84	0.64
ENS-EXP-SUB-EL-freq-gaze	∩	3.2 3B	×	✓	✓	0.75	0.42	0.52	0.89	0.51	0.65
		Naive	×	×	×	**0.75**	0.42	0.52	**0.89**	0.51	0.65
	∪	Naive	×	×	×	0.48	**0.75**	0.57	0.52	0.83	0.64
NER-EXP-SUB-1	∪	3.2 3B	×	✓	✓	0.48	0.75	0.58	0.51	0.84	0.63
		Naive	×	×	×	0.48	0.75	0.58	0.53	**0.84**	0.65
NER-EXP-SUB-EL-freq-gaze	∩	3.2 3B	×	✓	✓	0.74	0.43	0.52	0.87	0.51	0.65
		Naive	×	×	×	0.74	0.43	0.52	0.87	0.51	0.65
	∪	Naive	×	×	×	0.48	0.75	0.57	0.53	0.84	0.65

still relevant losses in NED.

Merged results The scores for the merges with ours and Graphwise in Tables 22 to 24 show similar heavy losses for our systems, with only strong precision remaining across all evaluated combinations. This precision helps our combined efforts achieve the most precise results in NERD. All tested top merges use naive enrichment and do not use RAG.

7.3. Named Entity Recognition and Disambiguation Results

The simple yet effective disambiguation pipeline based on TF-IDF in our system proved quite effective, as we observed only small drops in performance between the sub-subtasks of 6.1 and 6.2, i.e., the penalty for applying the disambiguation. Subtask 6.1, i.e., between NER and NERD, sees a drop of around 0.10 when comparing Tables 5 and 6, with the relative differences in Table 19. This translates to a drop of only 0.12 for the test set comparison in Tables 15 and 16, calculated in Table 20. Within Subtask 6.2 we observe a drop of about 0.01 in the best F_1 score on the development set (Tables 8 and 9), with a decrease of 0.02 in F_1 for the test set results of Tables 17 and 18, again calculated in Table 21. The test set for subtask 6.2, i.e., between RE and M-RE, has, in absolute terms, similar losses but suffers much higher relative losses due to the already low F_1 scores. Additionally, we observe much lower performance losses compared to the baseline for both subtasks.

Table 23
Top 10 Merged Test Set Results for NED (Subtask 6.1.2)

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
		Baseline	-	-	-	0.38	0.40	0.39	0.43	0.45	0.44
ENS-EXP-SUB-EL-freq-gaze	∩	3.2 3B	×	✓	✓	0.63	0.36	0.45	0.77	0.44	0.56
		Naive	×	×	×	**0.63**	0.36	0.45	**0.77**	0.44	**0.56**
	∪	3.2 3B	×	✓	✓	0.27	0.42	0.32	0.30	0.50	0.38
		Naive	×	×	×	0.26	0.42	0.32	0.31	0.50	0.39
NER-EXP-SUB-EL-freq-gaze	∩	3.2 3B	×	✓	✓	0.63	0.37	0.45	0.76	0.45	0.56
		Naive	×	×	×	0.63	0.37	**0.45**	0.76	0.45	0.56
	∪	3.2 3B	×	✓	✓	0.27	**0.43**	0.33	0.31	**0.50**	0.38
		Naive	×	×	×	0.27	0.43	0.32	0.32	**0.50**	0.39
gpt-5.shot-gold	∪	Naive	×	×	×	0.22	0.32	0.25	0.25	0.38	0.30

Table 24
Top 10 Merged Test Set Results for M-RE (Subtask 6.2.2)

Graphwise	Set	Model	Beams	NED FT	LoRA	P	R	F_1	P_{micro}	R_{micro}	$F_{1,micro}$
		Baseline	-	-	-	**0.10**	0.10	0.10	0.14	0.13	**0.13**
ENS-EXP-SUB-1	∪	3.2 3B	×	×	✓	0.03	0.04	0.03	0.06	0.07	0.06
ENS-EXP-SUB-EL-freq-gaze	∪	3.2 3B	×	×	✓	0.03	0.04	0.03	0.06	0.07	0.06
NER-EXP-SUB-1	∪	3.2 3B	×	×	✓	0.03	0.04	0.03	0.06	0.07	0.06
NER-EXP-SUB-EL-freq-gaze	∪	3.2 3B	×	×	✓	0.03	0.04	0.03	0.06	0.07	0.06
gpt-5.shot-gold	∩	3.2 3B	×	×	✓	0.05	0.01	0.01	**0.34**	0.02	0.04
	∪	3.2 3B	×	×	✓	0.07	**0.20**	**0.10**	0.09	**0.22**	0.12

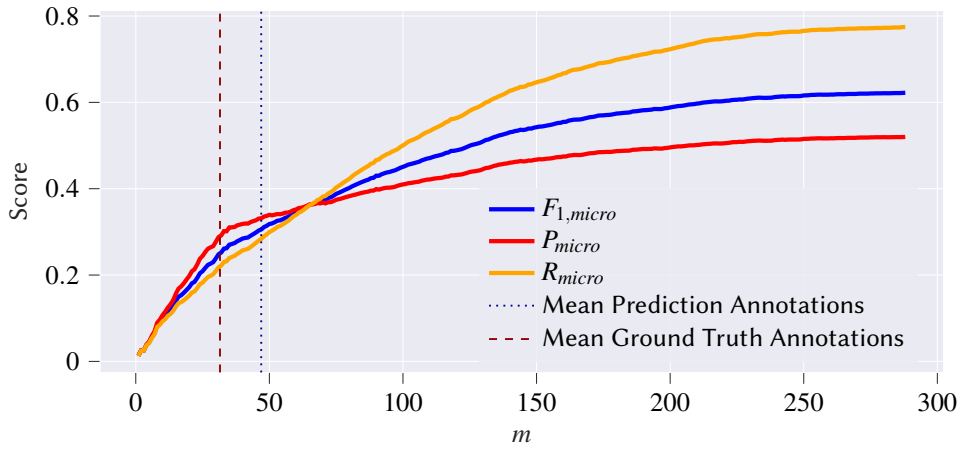


Figure 3: Scores over a varying m for NED on the Development Set using the NED finetune + Beam Serach to End + Naive Enrichment on the Hermes 3.1 8B model.

8. Conclusion & Discussion

CHASTE presents a novel combination of constrained generation and beam search for generative IE, specifically entity extraction and relation discovery. Based on this combination, we introduce two emerging extreme strategies,

(i.) a shallow full search over exponentially increasing combinations, and
(ii.) a depth-first search to the end of a prediction to evaluate the probabilities holistically.

This combination is embedded in a pipeline consisting of NED-informed, and ontologically enriched grammar, optionally combined with a naive, yet performant NED on prior, serving also as the baseline.

Using this novel combination, an evaluation campaign is conducted across a range of combinations, yielding insights into the specifics of the approach combinations. The first major insight is the strong naive baseline we compare against – in favor of the much stronger task baseline, which uses a very different approach. Nonetheless, we demonstrate that we can achieve a result beating our naive system on the development set *without* fine-tuning the full models using beam search or naive enrichment, and by fine-tuning only a relatively small tagging model. The result carries over to the test set, where we were unable to submit a result using beam search due to the prohibitively long runtime, and we also failed to deliver an effective shallow search. The non-fine-tuned model, nonetheless, dominates our other approaches – even while losing a lot of the performance we had on the development set. We, however, lose very little performance in the disambiguation step compared to the organizers' baseline, indicating that our naive TF-IDF approach remains very strong – albeit at a lower performance operating point.

Our combination of approaches with the Graphwise teams' method also revealed interesting insights in the combinational approaches, where we saw higher precision for the intersection, while the union operation boosted only recall at the cost of precision. This advantage led to our combined efforts achieving the best micro- and macro-precision on NERD, likely also due to our small losses when applying the disambiguation pipeline.

The introduction of beam search, at least the one we were able to evaluate, also indicates that it boosts precision while limiting the system's recall, at least for entity extraction. The RE, on the other hand, does not profit from the beam search, but rather loses some performance. This comparison is, notably, the first comprehensive evaluation of such a novel combination within a model's generative aspects, comparing fine-tuned models within a domain, RAG, to this holistic generative view.

Future Work We intend to improve the algorithmic aspects by re-evaluating our shallow beam search approach, tuning parameters, and possibly reducing inference time by pruning the tree earlier and skipping unlikely tokens by cutting probabilities earlier. The NED as an efficient prior could also benefit from improvements, as it places a hard cap on the maximum achievable performance, limiting us to scores below the baseline from the start.

Reproducibility

To ensure full reproducibility of our results, we supply the code as an artifact [22] derived from our GitHub Repository[3]. The repository contains the code and partial data to reproduce our results. The novel beam search approach is also in preparation for a forthcoming release of a Python package and will be complete for the camera-ready submission.

Acknowledgments

This work is partially supported by the HEREDITARY Project, as part of the European Union's Horizon Europe research and innovation programme under grant agreement No GA 101137074. We also would

[3] https://github.com/Dakantz/CHASTE

like to thank Marco Martinelli for the organizing and providing exceptional support in participating in the challenge.

Declaration on Generative AI

During the preparation of this work, the authors used Grammarly in order to: Grammar and spelling check and Github Copilot for enhanced autocompletion for writing the implementation. After using these tools, the authors reviewed and edited the code and content as needed and take full responsibility for the publication's content.

References

[1] R. J. Mooney, R. C. Bunescu, Mining knowledge from text using information extraction, SIGKDD Explor. 7 (2005) 3–10. doi:10.1145/1089815.1089817.

[2] B. Hoex, I. Razzak, J. Ren, Y. Wan, H. Wang, S. Wang, T. Xie, Y. Ye, W. Zhang, Construction and application of materials knowledge graph in multidisciplinary materials science via large language model, in: Advances in Neural Information Processing Systems 37, NeurIPS 2024, Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024, p. 56878–56897. doi:10.52202/079017-1812.

[3] L. Zhong, J. Wu, Q. Li, H. Peng, X. Wu, A comprehensive survey on automatic knowledge graph construction, ACM Comput. Surv. 56 (2024) 94:1–94:62. doi:10.1145/3618295.

[4] D. Xu, W. Chen, W. Peng, C. Zhang, T. Xu, X. Zhao, X. Wu, Y. Zheng, Y. Wang, E. Chen, Large language models for generative information extraction: a survey, Frontiers of Computer Science 18 (2024). doi:10.1007/s11704-024-40555-y.

[5] J. Dagdelen, A. Dunn, S. Lee, N. Walker, A. S. Rosen, G. Ceder, K. A. Persson, A. Jain, Structured information extraction from scientific text with large language models, Nature Communications 15 (2024). doi:10.1038/s41467-024-45563-x.

[6] B. Kantz, S. Lengauer, P. Waldert, T. Schreck, Constrained linked entity annotation using RAG (CLEANR), in: Working Notes of the Conference and Labs of the Evaluation Forum, CLEF 2025, Madrid, Spain, 9-12 September 2025, CEUR Workshop Proceedings, CEUR-WS.org, 2025, pp. 343–358. URL: https://ceur-ws.org/Vol-4038/paper_23.pdf.

[7] V. Steinbiss, B. Tran, H. Ney, Improvements in beam search, in: The 3rd International Conference on Spoken Language Processing, ICSLP 1994, Yokohama, Japan, September 18-22, 1994, ISCA, 1994, pp. 2143–2146. doi:10.21437/ICSLP.1994-538.

[8] J. E. Hu, H. Khayrallah, R. Culkin, P. Xia, T. Chen, M. Post, B. Van Durme, Improved lexically constrained decoding for translation and monolingual rewriting, in: Proceedings of the 2019 Conference of the North, Association for Computational Linguistics, 2019, p. 839–850. doi:10.18653/v1/n19-1090.

[9] M. Martinelli, V. Bonato, G. M. Di Nunzio, G. Faggioli, N. Ferro, O. Irrera, B. Kantz, S. Marchesin, L. Menotti, S. Merlo, R. Michelotto, G. Tussardi, F. Vezzani, P. Waldert, G. Silvello, Overview of GutBrainIE@CLEF 2026: Gut-Brain Interplay Information Extraction, in: E. S. Salido, A. Barrón-Cedeño, A. G. S. de Herrera, S. MacAvaney, J. M. Struß (Eds.), CLEF 2026 Working Notes, 2026.

[10] A. Nentidis, G. Katsimpras, A. Krithara, M. Krallinger, M. Rodríguez-Ortega, E. Rodriguez-López, N. Loukachevitch, I. Rozhkov, E. Tutubalina, D. Dimitriadis, G. Tsoumakas, G. Giannakoulas, A. Bekiaridou, A. Samaras, G. M. Di Nunzio, N. Ferro, S. Marchesin, M. Martinelli, G. Silvello, G. Paliouras, Overview of BioASQ 2026: The fourteenth BioASQ Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering, Lecture Notes in Computer Science, Springer, 2026.

[11] M. Schilling-Wilhelmi, M. Ríos-García, S. Shabih, M. V. Gil, S. Miret, C. T. Koch, J. A. Márquez,

K. M. Jablonka, From text to insight: large language models for chemical data extraction, Chemical Society Reviews 54 (2025) 1125–1150. doi:10.1039/d4cs00913d.

[12] B. Kantz, K. Innerebner, P. Waldert, S. Lengauer, E. Lex, T. Schreck, Graph queries from natural language using constrained language models and visual editing, in: 2025 IEEE International Conference on Knowledge Graph (ICKG), Limassol, Cyprus, November 13-14, 2025, IEEE, 2025, pp. 178–185. doi:10.1109/ICKG66886.2025.00030.

[13] M. X. Liu, F. Liu, A. J. Fiannaca, T. Koo, L. Dixon, M. Terry, C. J. Cai, "we need structured output": Towards user-centered constraints on large language model output, in: Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, CHI EA 2024, Honolulu, HI, USA, May 11-16, 2024, ACM, 2024, pp. 10:1–10:9. doi:10.1145/3613905.3650756.

[14] M. Post, D. Vilar, Fast lexically constrained decoding with dynamic beam allocation for neural machine translation, 2018. URL: https://arxiv.org/abs/1804.06609. arXiv:1804.06609.

[15] I. Montani, M. Honnibal, M. Honnibal, A. Boyd, S. V. Landeghem, H. Peters, explosion/spacy: v3.7.2: Fixes for apis and requirements, 2023. doi:10.5281/ZENODO.1212303.

[16] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter, CoRR abs/1910.01108 (2019). URL: http://arxiv.org/abs/1910.01108. arXiv:1910.01108.

[17] P. Deka, A. Jurek-Loughrey, D. P, Multiple evidence combination for fact-checking of health-related information, in: The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 237–247. URL: https://aclanthology.org/2023.bionlp-1.20.

[18] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, W. Chen, Lora: Low-rank adaptation of large language models, CoRR abs/2106.09685 (2021). URL: https://arxiv.org/abs/2106.09685. arXiv:2106.09685.

[19] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[20] R. Teknium, J. Quesnelle, C. Guang, Hermes 3 technical report, 2024. URL: https://arxiv.org/abs/2408.11857. arXiv:2408.11857.

[21] M. Martinelli, G. Silvello, V. Bonato, G. M. D. Nunzio, N. Ferro, O. Irrera, S. Marchesin, L. Menotti, F. Vezzani, Overview of gutbrainie@clef 2025: Gut-brain interplay information extraction, in: Working Notes of the Conference and Labs of the Evaluation Forum, CLEF 2025, Madrid, Spain, 9-12 September 2025, CEUR Workshop Proceedings, CEUR-WS.org, 2025, pp. 65–98. URL: https://ceur-ws.org/Vol-4038/paper_5.pdf.

[22] B. Kantz, Dakantz/chaste: Gutbrainie submission, 2026. doi:10.5281/ZENODO.20313871.