LlmLink: Dual LLMs for Dynamic Entity Linking on Long Narratives with Collaborative Memorisation and Prompt Optimisation

Lixing Zhu[*]
King's College London
lixing.zhu@kcl.ac.uk

Jun Wang[*]
University of Warwick
King's College London
jun.wang.3@warwick.ac.uk

Yulan He
King's College London
The Alan Turing Institute
yulan.he@kcl.ac.uk

Abstract

We address the task of CoREFerence resolution (CoREF) in chunked long narratives. Existing approaches remain either focused on supervised fine-tuning or limited to one-off prediction, which poses a challenge where the context is long. We develop a dynamic approach to cope with this: by deploying dual Large Language Models (LLMs), we assign specialised LLMs to local named entity recognition and distant CoREF tasks, respectively, while ensuring their exchange of information. Utilising our novel memorisation schemes, the coreference resolution LLM would memorise characters and their associated descriptions, thereby reducing token consumption compared with storing previous messages. To alleviate hallucinations of LLMs, we employ an automatic prompt optimisation method, with the LLM ranker modified to leverage annotations. Our approach achieves performance gains over other LLM-based models and fine-tuning approaches on long narrative datasets, significantly reducing the resources required for inference and training.

1 Introduction

When people read book-length narratives or episodic stories, they typically do not read the entire book in one sitting. Instead, they spread the reading over non-consecutive days. Each time they start a new chapter, they think of main characters and recall relevant descriptions associated with them, rather than revisiting earlier sections as if they were reading the book for the first time (Kintsch, 1994; Singer, 2017). Authors often facilitate this process by clearly marking divisions (e.g., Parts, Chapters, or Scenes) in their drafts, especially for long narratives such as novels, fiction and serial news reports. Moreover, humans comprehend long narratives using stratified processing: within a local context, readers concentrate on identifying narrative elements (e.g., characters, spatial locations and temporal specifications) (Piper et al., 2021), while in the global context, they attempt to link these characters to those stored in their memories (Brahman et al., 2021), thus updating their impressions on the characters after the reading (McDaniel et al., 2012).

However, recently developed models (Soni et al., 2023; Wagner et al., 2023) do not fully capture this entire process. In contrast, Jörke et al. (2020) used a dual attention model with BERT for local context and an upper attention layer for global context, but without storing past narrative elements. Fine-tuning LLMs pose a challenge due to quadratic scaling of Transformer attention. On the other hand, Wang et al. (2023) demonstrated that memorisation is helpful for text generation in the Project Gutenberg dataset, where context exceeds LLMs' capacities. Applying LLMs beyond their limited context remains challenging, even though evidence shows that LLMs are promising in entity linking (Hicke and Mimno, 2024; Zhang et al., 2023b). Resorting to in-context learning and expanding context by prepending conversation history would lead to high token consumption. It is inherently unnatural to start from the beginning of a book each time, despite already comprehending the prologue, as with extra-long context LLMs. Conversely, readers typically revisit earlier chapters to recall key characters before continuing with new content (Miyake and Shah, 1999). Mechanisms are needed to enable rewinding, enhancing character descriptions in memorisation.

Apart from memorisation, the dynamic processing of chunked narratives, such as books divided into chapters, is hindered because current state-of-the-art methods (Hicke and Mimno, 2024; Zhang et al., 2023b) rely on one-off prediction or pipeline approaches that combine both Named Entity Recognition (NER) and CoREFerence Resolution (CoREF). However, episodic reading of narratives is more common (Rayner et al., 2012). Bohnet et al. (2023) demonstrated that transition-based

[*]Equal contribution.

systems, which employ a link-append approach, are more capable of identifying anaphora (Webber et al., 2003) in long narratives.

To this end, we propose a joint entity recognition and coreference resolution model, called **LLMLINK**, which employs two instruction-tuned LLMs in a two-layered formation for handling short and long-term context, respectively. Motivated by recent work on narrative text generation (Wang et al., 2023), we introduce two memorisation schemes, *Prompt Cache* and *Conversation Memo*, to keep track of the characters and relevant plots. As our experiments show, both strategies integrate seamlessly with the proposed stratified architecture. The integrated model achieves significant improvements on two long narrative datasets and is more cost-efficient compared to simply expanding the LLM context.

A newly emergent challenge when using instruction-tuned LLMs is hallucination (Brahman et al., 2022), where LLMs exhibit undesirable behaviours such as incorrect entity span detection and ungrounded coreference resolutions. To address this, we adapt Automatic Prompt Optimisation (APO) (Pryzant et al., 2023), which optimises user prompts in the instruction based on available training examples. We modify the LLM ranker to rank our model's output according to ground-truth labels. By applying APO to LLMLINK, we obtain a competitive model that matches or surpasses purpose-built entity linking methods and memory-enhanced instruct LLMs, with lower token consumption and without the need for fine-tuning.

In summary, we demonstrate how instruction-tuned LLMs can jointly recognise named entities and resolve coreferences in dynamic settings for long narratives through memorisation and prompt optimisation. In particular:

1. We construct a dual LLMs framework that allocate distinct responsibilities to the models based on local and global contexts. This setup ensures that NER and coreference resolution are handled appropriately, with the upper-level LLM accessing to the lower-level information crucial for resolving coreferences.
2. We design memorisation strategies tailored for instruction-tuned LLMs and narrative understanding. These strategies not only enhance performance but also reducing token consumption significantly compared to methods relying on remembering previous messages.
3. We adapt APO and customise the LLM ranker to leverage NER and co-reference annotations for mitigating hallucinations commonly associated with LLMs.

2 Related Work

Coreference Resolution on Narratives Existing work on coreference resolution (Liu et al., 2019; Toshniwal et al., 2020a; Paolini et al., 2021; Zhang et al., 2022; Zheng et al., 2023) largely focused on short documents, e.g., MUC-7 (Hirschman and Chinchor, 1998), ACE (Doddington et al., 2004), OntoNotes (Hovy et al., 2006) and CoNLL-2011/2012 (Pradhan et al., 2011, 2012). On the other hand, Ravi et al. (2023); Ahmed et al. (2024); Nath et al. (2024); Min et al. (2024) addressed CoREF of news events on ECB+ (Ravi et al., 2023). However, few studies concentrated on narratives until Bamman et al. (2019) released LitBank. Toshniwal et al. (2020b); Baruah and Narayanan (2023) presented pipeline approaches of mention proposal and mention clustering. Jörke et al. (2020); Zhang et al. (2023b); Hicke and Mimno (2024) jointly detected mentions and coreferences in a single pass, while Xia et al. (2020); Bohnet et al. (2023); Guo et al. (2023) conducted joint NER and CoREF dynamically. Recent research interest (Hicke and Mimno, 2024) emerges in exploiting LLMs since context length is increased from 512 (Devlin et al., 2019) to 32K (Jiang et al., 2023). Despite their extended input context lengths, they are still inadequate for handling the narrative context due to the exponential cost or token consumption during the SFT or multi-turn inference. Additionally, these LLMs still struggle with fine-grained entity recognition and linking, often experiencing what is known as the "loss in the middle" phenomenon. We focus on addressing long-stride coreferences, where the context is so extensive that it exceeds the capacity of LLMs or incurs high computational costs. Contrary to the aforementioned approaches, we propose a hierarchical dual-LLM system augmented with the memorisation mechanism and the seamless integration of APO for joint NER and co-reference resolution on long narratives.

Automatic Prompt Optimisation Automatic prompt optimisation has recently been developed for refining prompts with annotated datasets or human feedback. This mechanism employs meta LLMs to analyse prompt quality and update prompts accordingly. Zhou et al. (2023) performed a Monte Carlo search over the semantic space of prompts based on LLM feedback. Zhang et al. (2023a) evaluated and edited prompts via reinforcement learning. Unlike the aforementioned work, Pryzant et al. (2023) developed a directed sampling method, which interprets feedback into semantic directions to guide the search for randomly sampled prompts. In contrast, Levi et al. (2024) employed a meta LLM (aka the Prompt Generator) to propose new prompts, whose instruction was generated by another meta LLM named Analyser. We adapt APO for optimising the user prompts in our dual LLMs by designing an additional meta LLM to evaluate the quality of prompts based on the NER and co-reference resolution results on long narratives.

3 LlmLink: Dynamic Entity Linking with Memorisation and Prompt Optimisation

We propose LlmLink, the **Dual LLM Memorisation and Optimisation model**, for entity recognition and linking within narrative understanding corpora (Zhu et al., 2023). This bears similarity with named entity recognition (NER) and coreference resolution (CR), as it involves identifying entities within text and linking references to the same entity across different mentions. We establish two instruction-LLM instances in a two-layered framework in which the lower layer identifies entities and the upper layer resolves coreferences.

We follow the InstructGPT (Ouyang et al., 2022) naming convention, where an **instruction** comprises a *system prompt* defining the task, a *user prompt* outlining the context-completion, and the *context* itself (which may include few-shot examples) while the LLM generates a completion. During training, an initial user prompt is crafted as the input to the lower LLM, with the system prompt fixed as *'You are an expert NER assistant. You are responsible for identifying named entities and generating their descriptions.'*. Meanwhile, the upper LLM is fed with a user prompt as defined in Section 3.1, with the system prompt set to *'You are an expert coreference resolution assistant. You are responsible for linking the entities to the resolved entities.'*.

We are interested in extracting and linking entities from long narratives, such as novels. A long narrative is segmented either by natural boundaries (such as parts, chapters, and sections) or into chunks of a typical maximum length. The lower LLM processes each chunk, wrapping the content with the user prompt and generating the completion (i.e., the tagged entities and their descriptions) in the format specified by the prompt. Subsequently, the completion text then splits into two dataflows: one for ranking the prediction against the ground-truth, and the other for forming the input to the upper LLM. The upper LLM receives the NER predictions along with auxiliary summaries from the lower LLM, and maintains a cache of some inputs and outputs. Using this cache, it generates predictions of coreferences that either link the entities in the current chunk to previously resolved ones or create new singletons.

The results are predicted coreference resolutions that are used for ranking to create incentives for APO. As we show in Figure 1, the training is a gradient-free process that optimises user prompts iteratively. In what follows, we elaborate on each component of the hierarchical framework and explain the significance of dual LLM, Cache, and APO.

3.1 Joint NER and Coreference Resolution with Dual LLMs

Dynamic entity linking on narrative text typically comprises two steps: each time a system encounters a text snippet, e.g., a paragraph, section or chapter, it first detects the named entities in the form of sub-strings of text, and next it resolves coreferences by marking each entity either as a mention or an emerging singleton. We limit the chunk size to less than the entire passage, thus presenting our model as a joint approach.

We follow the transition-based paradigm (Bohnet et al., 2023) to link the entities incrementally, in contrast to the one-off prediction of all clusters of mentions in a single pass (Zhang et al., 2023b). We propose to **deploy two LLMs in a collaborative setup**. Consider a narrative $\boldsymbol{x_i}$ and its constitutional chunks $\{x_{i1}, x_{i2}, \ldots, x_{im}\}$, where x_{ij} is the j-th chunk. The input to *the lower layer LLM* (called **NER LLM**) is a concatenation of the user prompt and the text chunk. An example for $j = 1$ in Figure 1 is shown below:

User prompt u_{L1}, see Appendix A for full prompts.

Input : $\overbrace{\text{Given a chapter, you have to identify}}$ ONLY the following: PROPER_NOUN - The full or partial name of an individual person, place, or organization . . . NOUN_PHRASE - A grouping of words that includes a noun, and functions within a sentence as a subject, object, or another role, . . . PRONOUNS - A word that is used instead of a noun or noun phrase. This includes common forms (*I, me . . . their*), historical forms (*thou,thee . . . ye*) and forms originating in transcriptions of speech (*'em, 'ee, yeh, yer*). You will output in JSON format. . . The input is: PART ONE–The

The content of the first chapter.

Old Buccaneer\n 1\n The Old Sea-dog . . . to come.

The output of the NER LLM is a structure comprising entities and auxiliary descriptions (see Appendix B for formats) denoted as $\{\boldsymbol{e}_{ij}, \boldsymbol{a}_{ij}\}$.

The *upper layer* is a new instance of LM which we call the **resolver LLM**, whose input is u_H appended with $\boldsymbol{r}^*_{ij}, \boldsymbol{d}^*_{ij}, \boldsymbol{e}_{ij}, \boldsymbol{a}_{ij}, x_{ij}$ where $\boldsymbol{r}^*_{ij}$ is an ordered dictionary of resolved entities that each key refers to unique character or place. $\boldsymbol{d}^*_{ij}$ are auxiliary descriptions of the appeared singletons. It starts with an empty cache at $j = 1$. The user prompt formalises the ingredients into an instruction as shown below:

User prompt u_H

Input : Given the resolved entities and their auxiliary descriptions, a chapter and the identified entities within the chapter, you have to do one of the following operations to every identified entity: (1) link the entity to one of the resolved entities if they mention the same thing; (2) create a new entity type and mark it as singleton if it is a PROPER_NOUN and cannot be linked to any resolved entity; (3) mark the entity as None if the entity can neither be marked as singleton nor linked . . . The input is:

The JSON object of $\{\boldsymbol{r}^*_{i1}, \boldsymbol{d}^*_{i1}, \boldsymbol{e}_{i1}, \boldsymbol{a}_{i1}, x_{i1}\}$.

{"RESOLVED_ENTITIES": null, "RESOLVED_ENTITIES_DESCRIPTION": null, . . . }

The output, $\{\boldsymbol{r}_{ij}, \boldsymbol{d}_{ij}\}$, consists of a map of entities with newly discovered singletons and new links between the mentions and the proper nouns, presented in JSON strings. It also includes a dictionary of the updated descriptions of the entities, which will subsequently update prior descriptions, as will be discussed in Section 3.2.

3.2 Cache in Prompt vs. Conversation History

LlmLink achieves memorisation (Wang et al., 2023) via two mechanisms we propose — prompt cache and conversation memo. This is distinct from the previous work (Zhang et al., 2023b; Hicke and Mimno, 2024) in which memorisation is implemented by expanding the context to the entire document.

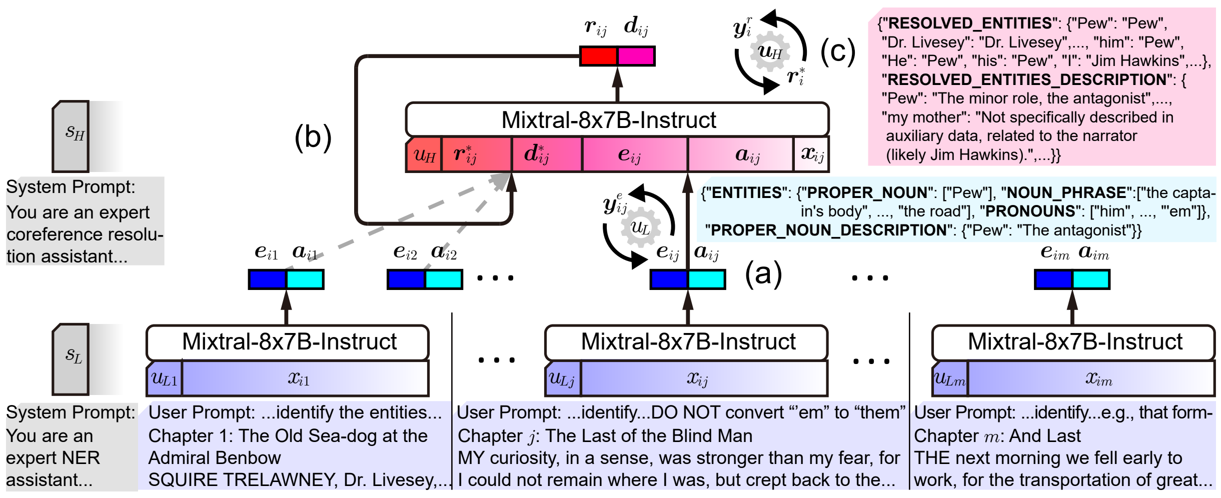
Prompt Cache is a JSON-string-format dictionary which records the coreference clusters. The rationale is that coreference relations are directed acyclic graphs (Webber et al., 2003) that can be stored in an adjacent table and applied to streaming data, i.e., document chunks. At the j-th operation, the prompt cache is defined as follows: $\boldsymbol{r}^*_{ij} = \{(k, v) | k \in \mathit{proper\_noun\_set}_{ij}, v \in \mathit{noun\_phrase\_set}_{ij} \cup \mathit{pronoun\_set}_{ij}\}$. The resolver LM $\boldsymbol{r}_{ij}$ updates $\boldsymbol{r}^*_{ij}$ to $\boldsymbol{r}^*_{i,j+1}$ with newly discovered anaphors $\boldsymbol{r}_{ij}$. The auxiliary descriptions are updated accordingly.

Conversation Memo does not explicitly store the resolved entities in any structure. Instead, it keeps track of the conversation history. After the j-th prediction, $\{\boldsymbol{e}_{ij}, \boldsymbol{a}_{ij}, \boldsymbol{r}_{ij}, \boldsymbol{d}_{ij}\}$ is appended to the conversation memo, which serves as an assistant message input to the resolver LM in the next step. We expand auxiliary descriptions to include a story summary in addition to descriptions of characters and places. To comply with the prompt cache format and steer the generation, we initialise $\{\boldsymbol{r}^*_{ij}, \boldsymbol{d}^*_{ij}\}$ as $\{\boldsymbol{r}_{ij}, \boldsymbol{d}_{ij}\}$ and start the recording from $j = 2$.

Algorithm 1: LlmLink for document $\boldsymbol{x}_i$

```
Input: A chunked document $x_i = \{x_{i,1:m}\}$,
       system prompts $\{s_L, s_H\}$, seed user
       prompts $\{u_{L1}, u_H\}$, $maxRewind$.
Output: A dictionary $r^*_i$ of recognized
        entities and their coreferences.
1  for $t \leftarrow 1$ to $maxRewind$ do
2      $j \leftarrow 1$
3      if $t = 1$ then
4          $\{r_{i1}, d_{i1}\} \leftarrow$ resolverLlm($e_{i1}, a_{i1}, x_{i1}$)
5          $r^*_{i1} \leftarrow r_{i1}$, $d^*_{i1} \leftarrow d_{i1}$, $j \leftarrow 2$
6      else
7          $r^*_{i0} \leftarrow r^*_{im}$, $d^*_{i0} \leftarrow d^*_{im}$
8      while $j \leq m$ do
9          if $isPromptCache$ then
10             $\{r_{ij}, d_{ij}\} \leftarrow$ resolverLlm($r^*_{i,j-1}, d^*_{i,j-1}, e_{ij}, a_{ij}, x_{ij}$)
11         else
12             $\{r_{ij}, d_{ij}\} \leftarrow$ resolverLlm($h_{i,j-1}, r^*_{i,1}, d^*_{i,1}, e_{ij}, a_{ij}, x_{ij}$)
13         $r^*_{ij} \leftarrow$ update($r^*_{i,j-1}, r_{ij}$)
14         $d^*_{ij} \leftarrow$ update($d^*_{i,j-1}, d_{ij}$)
15         $j \leftarrow j + 1$
16 return $r^*_i$
```

Let $\boldsymbol{h}_{ij}$ denote the conversation history. The entities and coreferences of a narrative are identified using a scan through the document dis-



{"**RESOLVED_ENTITIES**": {"Pew": "Pew", "Dr. Livesey": "Dr. Livesey",..., "him": "Pew", "He": "Pew", "his": "Pew", "I": "Jim Hawkins",...}, "**RESOLVED_ENTITIES_DESCRIPTION**": { "Pew": "The minor role, the antagonist",..., "my mother": "Not specifically described in auxiliary data, related to the narrator (likely Jim Hawkins).",...}}

System Prompt: You are an expert coreference resolution assistant...

{"**ENTITIES**": {"**PROPER_NOUN**": ["Pew"], "**NOUN_PHRASE**":["the captain's body", ..., "the road"], "**PRONOUNS**": ["him", ..., "'em"]}, "**PROPER_NOUN_DESCRIPTION**": {"Pew": "The antagonist"}}

System Prompt: You are an expert NER assistant...

User Prompt: ...identify the entities... Chapter 1: The Old Sea-dog at the Admiral Benbow SQUIRE TRELAWNEY, Dr. Livesey,...

User Prompt: ...identify...DO NOT convert "'em" to "them" Chapter j: The Last of the Blind Man MY curiosity, in a sense, was stronger than my fear, for I could not remain where I was, but crept back to the...

User Prompt: ...identify...e.g., that form- Chapter m: And Last THE next morning we fell early to work, for the transportation of great...

Figure 1: The system diagram of LLMLINK. **(a) A lower-level LLM** is deployed with a system prompt and user prompts for NER. A text chunk is fed into the NER expert model, which identifies entities and generates auxiliary descriptions. **(b) An upper-level LLM**, specialised in coreference resolution, will then process those entities and descriptions and output resolved entities. The upper-level LLM builds its memory from previous conversations either by enriching the user prompt or supplying the assistant message. **(c) Automatic Prompt Optimisation (APO)** is employed as the optimiser for training with named entity labels and coreference links.

played in Algorithm 1. Both memorisation approaches can be used interchangeably by switching the $isPromptCache$ configuration. To tackle the flashback that a character is mentioned earlier in the narrative than its proper noun, we use an extra loop that rewinds a document at least once, during which the resolver LLM inherits the memory of the previous scan.

3.3 Automatic Prompt Optimisation

Algorithm 1 is a zero-shot approach for joint entity recognition and linking. Although effective, the model sometimes lacks alignment with the narrative context. For example, the NER LLM tends to convert eye dialects to their norm forms deliberately, and the resolver LLM occasionally omits backtracks in single blocks. To address these issues, additional efforts are needed to reduce hallucinationsand improve robustness.

While supervised fine-tuning is promising when annotations are available, the extensive context size and fragmented error cases make it impractical. In contrast, Automatic Prompt Optimisation (APO) (Pryzant et al., 2023) provides a gradient-free alternative to address the misalignment between annotations and user intentions, fully leveraging LLM capabilities without jeopardising other tasks. We propose adapting AutoPrompt (Levi et al., 2024) to optimise and update are the user prompt for the NER LLM, u_L, and the user prompt for the resolver LLM, u_H. AutoPrompt is an external plugin which utilises **three meta LLMs** designed to generate challenging samples (aka the **Sample Generator**), provide editing suggestions (aka the **Analyser**), and upgrade prompts (aka the **Prompt Generator**), respectively. We do not use the Sample Generator, as we rely on the dataset for the ground truth and data points. But we introduce an additional meta LLM as the **LLM Ranker** to evaluate the quality of the prompts, which is invoked before the Analyser.

LLM Ranker: Prompt optimisation requires evaluating the quality of the generated prompts. We use specialised LLM rankers to perform the ranking, following the prompt template/guidance in (Levi et al., 2024) on converting a text generation prompt into a ranking prompt. For judging predictions of named entities from the NER LLM (i.e., the lower-level LLM), we propose a 5-scale ranking based on the percentage of correct predictions relative to the ground truth (full prompt detailed in Appendix C). For the resolver LLM (i.e., the upper-level LLM), we use another customised LLM specified by a ranker inductive prompt, which is modified to '*...assigning the classification labels...distinctly representing the accuracy of predicted coreferences...*' (see Appendix C). The LLM ranker uses coreference ground-truth labels to compute the ranks.

Optimising u_L: In the inner loop of Algorithm 1, APO is executed at each iteration, immediately after the identification of e_{ij}. This results in a sequence of updated prompts $\{u_{L,1}, u_{L,2}, \ldots, u_{L,m}\}$. Note that APO is also invoked during the final rewind to achieve the optimal NER performance.

We use the ‘Generative Tasks’ setup of AutoPrompt since s_L and u_L instruct the LLM to function as a text generator. Other meta LLMs in APO such as Analyser and Prompt Generator are kept as default.
Optimising u_H: We place the APO procedure in the final iteration. Similar to u_L, the APO runs within the ‘Generative Tasks’ setup.

Other hyperparameters, such as the number of updates (i.e., how many times the model iterates over the whole dataset), are discussed in the Experiments section.

4 Experimental Setup

4.1 Datasets

Our dynamic entity linking approach is evaluated on the Coreference-Annotated LitBank (Bamman et al., 2020) and the refactored NarrativeQA (Kočiský et al., 2018) dataset. The Coreference-Annotated LitBank dataset[1] is a fiction dataset with entities annotated in the ACE format. Coreferences are resolved for proper names (PROP) (e.g., *Dr. Livesey*), common phrases (NOM) (e.g., *his room*), and pronouns (PRON) (e.g., *him*). Each coreference is grounded to a singleton (i.e., the first occurrence of that entity in the coreference chain). We followed Bamman et al. (2020); Zhang et al. (2023b); Guo et al. (2023) [2] and divided the dataset into 90 documents for training and 10 documents for testing.[3]. Duplicate annotations (e.g., lines 475 and 476 in ‘*27_far_from_the_madding_crowd_brat.ann*’) were removed. For the NarrativeQA dataset[4], we followed the BookQA approach (Angelidis et al., 2019) to refactor a subset of *Who* questions that target characters as answers. We excluded *Who does* questions and included *What’s the name* questions. We further filtered out questions where the named entity or the named entity of the answer cannot be grounded to a text span in the original text, ensuring that each question defines a long-term coreference. This results in 401 annotations that ground references to singleton characters. We refer to this subset of the dataset as WhoLink. The number of documents and statistics for each dataset are listed in the Table 1.

[1]Available as part of LitBank at https://github.com/dbamman/litbank/tree/master/coref

[2]https://github.com/dbamman/lrec2020-coref

[3]Validation was not performed since early stopping was not used; instead, prompts were optimised for a fixed number of rounds as in (Levi et al., 2024).

[4]https://github.com/google-deepmind/narrativeqa/tree/master

[5]https://github.com/somethingx1202/LlmLink

Datasets	# Docs		# Chapters		# Tokens		# Mentions	
	Train	Test	Train	Test	Train	Test	Train	Test
LitBank	90	10	25.06	25.06	102k	102k	291	291
WhoLink[5]	133	100	81.87	54.75	177k	131k	3.76	2.86

Table 1: Training and testing set statistics include document count, average chapters, tokens, and mentions per document. Note that LitBank has been presented with splits for 10-fold cross-validation.

4.2 Baseline Models

We compare our method against these baselines:
LCA-LLM (Hicke and Mimno, 2024) is a fine-tuned T5 (Raffel et al., 2020), a multi-task text-to-text framework in which the most effective model utilises both a Transformer encoder and decoder. We followed the setup of (Hicke and Mimno, 2024) to fine-tune T5 as a causal LM on the two datasets.

DOC-ARC (Jörke et al., 2020) is a hierarchical dual-attention model comprising two layers of attention, with the lower-level attention using a pre-trained language model for token representations, while the higher-level attention overseeing all the occurrences of selected tokens. Embeddings of the same entity type are aggregated using weights calculated by an attention mechanism.
Pure-Seq2seq (Zhang et al., 2023b) is an unmodified T0 (Sanh et al., 2022) model fine-tuned with coreference annotations in a text generation setup. Annotations were converted into natural language text by wrapping them in coreference cluster formats with special tokens, and the fine-tuning of T0 is formulated as completing the token sequence.
Dual-Cache Guo et al. (2023) introduces a global cache, managed by Longformer (Beltagy et al., 2020), for tracking long-distance mentions alongside an off-the-shelf local mention detector. When a mention is detected, the model uses a Multi-layer Perceptron to compute its similarity scores against every cached entity and links the mention to the closest singleton.
MovieCoref, developed by Baruah and Narayanan (2023), adapts the word-level CR model of Dobrovolskii (2021). This model first identifies a head word, then links words based on coreference scores. Finally, each linked word is extended to an entity span by identifying the maximum start-end scores in their neighbourhoods in conjunction with other words in the coreference chain.

4.3 Settings

Each book is rewound twice ($maxRewind = 2$), and prompts are optimised during the second pass.

LitBank			
Model	NER F1	Coref. F1	Exact String Match
LCA-LLM	97.9	80.2	71.1
DOC-ARC	93.9	76.1	67.4
Pure-Seq2seq	97.2	78.2	68.7
Dual-Cache	97.3	79.8	70.4
MovieCoref	92.4	74.9	67.5
Mixtral8x7bIns.	96.7	77.4	69.1
GPT-3.5-turbo	95.6	75.3	68.2
LLMLINK	**98.4**$\pm.2$	**81.5**$\pm.4$	**73.0**$\pm.5$
WhoLink			
LCA-LLM	89.2	71.4	64.7
DOC-ARC	85.0	59.8	54.3
Pure-Seq2seq	87.8	68.5	62.5
Dual-Cache	88.1	69.1	63.1
MovieCoref	83.2	56.6	53.8
Mixtral8x7bIns.	**91.6**	75.6	68.7
GPT-3.5-turbo	90.6	74.8	68.2
LLMLINK	**91.6**$\pm.3$	**78.2**$\pm.5$	**71.6**$\pm.5$

Table 2: Results for named entity recognition and all coreference resolution (pron/non-pron), reported in %.

The training set is iterated 5 times. One key configuration is how chapters are defined, as this determines the context size, for which we design regular expressions. For documents without chapter boundaries, we segment them into chunks of $8,192$ tokens, which results in statistics shown in Table 1. The maximum context size, including the auxiliary descriptions, is $16,384$ tokens. Excessive descriptions of NOMs and PRONs are trimmed to comply with the context limit. In conversation memos, earlier conversations are truncated once the upper LLM reaches its capacity. We modify Mixtral-8x7B-Instruct-v0.1[6] (Jiang et al., 2024) for implementation. Temperature and top_p are set to 0. Our APO module adapts AutoPrompt[7] implementation, using the generation configuration and the GT ranker as GPT-4 Turbo.

5 Experimental Results

LLMLINK is built with instruction-tuned LLMs. Hence, it operates in a free-form completion mode where 'prediction' of hallucinated named entities or coreferences becomes unavoidable. Employing traditional coreference metrics (i.e., MUC, B^3, CEAF and BLANC) (Pradhan et al., 2014) will result in low precision. Therefore, we resort to F1-scores of named entity detection and coreference resolution, as in (Hicke and Mimno, 2024; Guo et al., 2023). Exact string matches refer to instances where character positions match those in the original literature.

5.1 Overall Comparison

Results in Table 2 show that LCA-LLM and Mixtral8x7BInstruct are the top-performning baselines in both NER and Coreference Resolution. This is not surprising since LCA-LLM benefits from fine-tuning, while Mixtral8x7BInstruct contains 56 billion parameters. The key difference is that LCA-LLM predicts token logits and has to be fine-tuned on the training set, potentially explaining the performance gap observed between these two models on LitBank and WhoLink, where annotations in WhoLink are more sparse. Their Exact String Match results across the datasets indicates that token tagging models tend to be more stable.

Among the text completion models, Pure-Seq2seq performs the best on LitBank but the worst on WhoLink, showing its specialised nature. Mixtral demonstrates a significant advantage over Pure-Seq2seq on WhoLink due to its instruction-tuned design. Our model, LLMLINK, achieves the best overall results on both datasets, except for NER-F1 on WhoLink where it performs equally well. On the other hand, Dual-Cache outperforms DOC-ARC, showing a gain of at least 3% F1-score on LitBank and $\simeq 10\%$ gain on WhoLink. Meanwhile, DOC-ARC surpasses MovieCoref by at least 1% F1-score. These findings demonstrate the critical role of memorisation, especially in handling long context. Overall, instruction-tuned models, such as Mixtral and GPT, excel on WhoLink with consistent performance. Token tagging models achieve higher performance when fine-tuned with sufficient annotations. Text completion models, i.e., Pure-Seq2seq, benefit from fine-tuning, but they are less competitive compared to models like LCA-LLM, possibly due to the lack of instruction-tuning. LLMLINK outperforms LCA-LLM primarily due to its access to annotations.

5.2 Ablations

To analyse computational costs, we report token consumption data for instruction-tuned models and ablated components in Table 3. While token consumption remains relatively low during inference for other baselines (e.g., 11.27M for LCA-LLM,

[6] https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1

[7] https://github.com/Eladlev/AutoPrompt

Model	LitBank		WhoLink	
	Coref. F1	Cumul. Token Consum.	Coref. F1	Cumul. Token Consum.
M.8x7bIns.	77.4±.1	11.78M	75.6±.4	42.85M
+ Memo	80.6±.1	69.17M	77.2±.4	417.33M
Dual-LLMs	77.9±.2	22.56M	75.8±.4	74.83M
+ Cache	80.3±.2	32.13M	77.6±.4	127.41M
+ Memo	79.5±.2	35.21M	76.9±.4	134.28M
LlmLink	**81.5**±.4	32.73M	**78.2**±.5	129.52M
w/o APO	80.6±.2	31.70M	78.1±.4	118.53M
w/o LLM ranker	79.3±.4	31.90M	76.0±.5	122.72M

Table 3: Coreference F1 score and cumulative token consumption for LlmLink and examined architectures, with and without ablated components.

10.29M for DOC-ARC, and 21.51M for Pure-Seq2seq), the cost of fine-tuning is significantly higher[8], rendering direct comparison infeasible here.

Table 3 shows the performance gains through various frameworks and components. Deploying the Dual-LLM structure, i.e., simply using an additional LLM for coreference resolution without memorisation, yields a minor boost of $\simeq 0.5\%$. Despite the marginal gain, it serves as a foundation for the two memorisation designs. We obtain major improvements with them on top of the Dual-LLM architecture: introducing Cache brings a boost of 2.4%, and incorporating Memo yields an increase of 1.6%. The improved model costs less than 35.21M tokens, which is around half the token consumption of Mixtral-with-Memo. The improvement is higher if we choose to alternate between Cache and Memo. Utilising APO further enhances the model, yielding a performance gain of around 1% on LitBank. It is worth noting that, for Mixtral8x7BInstruct, its memorisation size is restricted by its 32K context size. If the maximum context size were increased to 100K, the token consumption would increase quadratically as the context accumulates over subsequent chapters' conversations. We also notice that without using our proposed LLM ranker for APO, the performance degrades significantly,

5.3 Case Study

We study cases where the proposed memorisation mechanisms or the introduced APO process make a difference in the predicted labels:

[8]https://openai.com/api/pricing/

Named Entity Recognition
Input: (see Appendix D for full text)

```
At that another fellow, probably him who had...
```

Output:

```
{"ENTITIES":{"PROPER_NOUN":["Pew"],"NOUN_PHRASE
": ["the captain's body","the door of the inn",
"that boy","the blind man","the fellow from the
window"], "PRONOUNS": ["him", "he", "I", "his",
"they", "I", "I", "them", "they", "them"]},
"PROPER_NOUN_DESCRIPTION": {"Pew": "A character
, specifically referred to as a blind man"}}
```

Output (after APO):

```
{"ENTITIES": {"PROPER_NOUN": ["Bill", "Pew"], "
NOUN_PHRASE":["the captain's body","the door of
the inn", "the inn", "that boy", "the blind man
","the road"], "PRONOUNS": ["him","he","I","his"
, "they", "'em", "they", "'em"]},
"PROPER_NOUN_DESCRIPTION": {"Bill": "A character
possibly related to a past event", "Pew":"The b-
lind man character in the narrative"}}
```

As illustrated above, LlmLink erroneously converted '*'em*' to '*them*', which does not exist in the original text. During APO, an error analysis, i.e., '*Failed to recognize "'em" which is a colloquial form of "them"*', is generated, and the user prompt is updated accordingly, enabling the generation of the correct output.

Coreference Resolution
Input: (see Appendix E for full text)

```
{"RESOLVED_ENTITIES": {"Tom Sawyer": "singlet...
```

Output:

```
{"Tom":"Tom Sawyer", "Huckleberry":"Huckleberry
Finn", "Muff Potter":"Muff Potter","the far end
of the graveyard":null,"their":null,"they":null
,"them":null,"we":null,"you":null,"us":null,"I"
:null,"me":null,"you":null,"them":null}
```

Output (with Memo "Input: {"IDENTIFIED_ ENTITIES": ..., {"PROPER_NOUN_ DESCRIPTION": ...}, Output: {"Tom": "Tom Sawyer", "Sid": "Sid", ..., "he": "Tom Sawyer"}"):

```
{"Tom": "Tom Sawyer","Huckleberry": "Huckleberry
Finn","Muff Potter": "Muff Potter", "the far end
of the graveyard": null,"their": "Tom Sawyer and
Huckleberry Finn", "they": "Tom Sawyer and Huck-
leberry Finn","them":"Tom Sawyer and Huckleberry
Finn", "we": "Tom Sawyer and Huckleberry Finn",
"you": "Huckleberry Finn", "us": "Tom Sawyer and
Huckleberry Finn", "I": "Tom Sawyer", "me": "Tom
Sawyer", "you": "Huckleberry Finn", "them": "The
devils or humans"}
```

We can observe nullified resolutions of the entities, i.e., the pronouns, in the initial output. By using previous conversations, the model successfully links the pronouns to the corresponding characters.

6 Conclusion

We have developed a joint named entity recognition and coreference resolution model that employs dual LLMs to process segmented long narratives. A lower-level LLM is placed within the chapter-wide segment for NER, whose output is consumed by an upper-level LLM, which resolves entities to the same character, i.e., singleton, simultaneously. To increase accuracy while lowering cost, we explore Cache and Memo as memorisation schemes, which fit comfortably with instruction LLMs. What's more, we utilise Automatic Prompt Optimisation with the LLM ranker to exploit training instances which are scattered and were erroneously predicted. The integrated model shows compelling performance against fine-tuned models and memory-enhanced instruct-LLMs, significantly reducing the amount of token consumption.

Limitations

While LLMLINK enjoys the benefit of in-context learning that is suitable for zero-shot and few-shot prediction, it inherits the weaknesses of instruction-tuned LLMs that extra tokens are needed and prompt engineering is intricate. On the contrary, hidden representation models such as BERT consume fewer tokens once fine-tuned and are unlikely to generate hallucinations or perform inconsistently. Another limitation of using dual LLMs is the design of collaborative pattern, i.e., the lower layer LLM receives no signals from the upper layer, which we do not optimise. Additionally, $maxRewind$ and rewind timing are preset ahead of the inference. It remains a challenge how to shift focus backwards during the interim of the reading process.

Acknowledgements

This work was supported in part by the UK Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Fellowship (grant no. EP/V020579/1, EP/V020579/2), and Innovate UK through the Accelerating Trustworthy AI programme (grant no. 10093055).

References

Shafiuddin Rehan Ahmed, George Arthur Baker, Evi Judge, Michael Reagan, Kristin Wright-Bettner, Martha Palmer, and James H. Martin. 2024. Linear cross-document event coreference resolution with X-AMR. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 10517–10529, Torino, Italia. ELRA and ICCL.

Stefanos Angelidis, Lea Frermann, Diego Marcheggiani, Roi Blanco, and Lluís Màrquez. 2019. Book QA: Stories of challenges and opportunities. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 78–85, Hong Kong, China. Association for Computational Linguistics.

David Bamman, Olivia Lewke, and Anya Mansoor. 2020. An annotated dataset of coreference in English literature. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 44–54, Marseille, France. European Language Resources Association.

David Bamman, Sejal Popat, and Sheng Shen. 2019. An annotated dataset of literary entities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2138–2144, Minneapolis, Minnesota. Association for Computational Linguistics.

Sabyasachee Baruah and Shrikanth Narayanan. 2023. Character coreference resolution in movie screenplays. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10300–10313, Toronto, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Bernd Bohnet, Chris Alberti, and Michael Collins. 2023. Coreference resolution through a seq2seq transition-based system. *Transactions of the Association for Computational Linguistics*, 11:212–226.

Faeze Brahman, Meng Huang, Oyvind Tafjord, Chao Zhao, Mrinmaya Sachan, and Snigdha Chaturvedi. 2021. "let your characters tell their story": A dataset for character-centric narrative understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1734–1752, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Faeze Brahman, Baolin Peng, Michel Galley, Sudha Rao, Bill Dolan, Snigdha Chaturvedi, and Jianfeng Gao. 2022. Grounded keys-to-text generation: Towards factual open-ended generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7397–7413, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.