# Empirical Study of Zero-shot Keyphrase Extraction with Large Language Models

**Byungha Kang and Youhyun Shin***
Department of Computer Science and Engineering, Incheon National University
{bhkang, yhshin}@inu.ac.kr

## Abstract

This study investigates the effectiveness of Large Language Models (LLMs) for zero-shot keyphrase extraction (KE). We propose and evaluate four prompting strategies: vanilla, role prompting, candidate-based prompting, and hybrid prompting. Experiments conducted on six widely-used KE benchmark datasets demonstrate that Llama3-8B-Instruct with vanilla prompting outperforms state-of-the-art unsupervised methods, PromptRank, by an average of 9.43%, 7.68%, and 4.82% in F1@5, F1@10, and F1@15, respectively. Hybrid prompting, which combines the strengths of vanilla and candidate-based prompting, further enhances overall performance. Moreover role prompting, which assigns a task-related role to LLMs, consistently improves performance across various prompting strategies. We also explore the impact of model size and different LLM series: GPT-4o, Gemma2, and Qwen2. Results show that Llama3 and Gemma2 demonstrate the strongest zero-shot KE performance, with hybrid prompting consistently enhancing results across most LLMs. We hope this study provides insights to researchers exploring LLMs in KE tasks, as well as practical guidance for model selection in real-world applications. Our code is available at https://github.com/kangnlp/Zero-shot-KPE-with-LLMs.

(a) Vanilla

```
Extract keyphrases from the text. The answer should be listed after
'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1
; keyphrase 2 ; ... ; keyphrase N'"

Text: {document}
```

(b) Role Prompting

```
You are a keyphrase extractor. Extract keyphrases from the text. The
answer should be listed after 'Keyphrases: ' and separated by semicolons
(;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'"

Text: {document}
```

(c) Candidate-based Prompting

```
You are a keyphrase extractor. Extract top 5 keyphrases from the 'Key
phrase candidates' consisting of noun phrases extracted from the text.
The answer should be listed after 'Keyphrases: ' and separated by semic
olons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'

Text: {document}
Keyphrase candidates: {Candidates (noun phrases)}
```

(d) Hybrid Prompting

```
You are a keyphrase extractor. Extract top 15 keyphrases from the 'Key
phrase candidates'. The answer should be listed after 'Keyphrases: ' and
separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ...
; keyphrase N'

Text: {document}
Keyphrase candidates: {(b)Output + (c)Output}
```

Figure 1: Examples of four prompting strategies for zero-shot keyphrase extraction.

## 1 Introduction

Keyphrases are single words or multi-word phrases that summarize the core content of a document. Keyphrase extraction (KE) is the task of automatically identifying and extracting multiple keyphrases from a given document. Numerous KE methods have been proposed, with a significant focus on unsupervised approaches due to their flexibility and applicability across diverse domains. Most unsupervised keyphrase extraction (UKE) methods follow a two-step process: 1) extracting candidates from the document and 2) estimating their importance. The candidate set is constructed by performing part-of-speech (POS) tagging on all words in the given document and extracting all noun phrases, resulting in a larger number of candidates than the ground truth keyphrases. Thus, accurately estimating the importance of each candidate is crucial for the performance of UKE. To estimate the importance of candidates, various methods have been proposed, including statistical (Sparck Jones, 1988; Campos et al., 2018), graph-based (Mihalcea and Tarau, 2004; Wan and Xiao, 2008), and embedding-based (Bennani-Smires et al., 2018; Sun et al., 2020; Liang et al., 2021; Zhang et al.,

* Corresponding author.



*Proceedings of the 31st International Conference on Computational Linguistics*, pages 3670–3686
January 19–24, 2025. 

2022; Song et al., 2023) approaches.

The emergence of Large Language Models (LLMs), such as ChatGPT[1] and Llama (Touvron et al., 2023), has transformed the NLP landscape. These instruction-following LLMs can perform various NLP tasks in a zero-shot setting simply by providing task instructions as prompts. Consequently, a new approach to UKE has become possible, which guides LLMs to generate a sequence of correct keyphrases by providing an appropriate prompt as context. This ***prompting-based*** approach is distinctly different from traditional methods that focused on measuring the importance of candidates. Given the potential of this novel approach, this study aims to explore and leverage the capabilities of LLMs for zero-shot KE. We also investigate whether LLMs possess the ability to estimate candidate importance through prompting. *Which is more effective: directly instructing LLMs to extract keyphrases, or instructing LLMs to select keyphrases from a set of candidates? Additionally, are there ways to improve the zero-shot performance of LLMs? Can we achieve improved performance by combining these approaches?* To address these questions and thoroughly explore the potential of LLMs for KE, we propose and evaluate four prompting strategies:

- **Vanilla**: We evaluate the fundamental KE capabilities of LLMs, acquired through their pre-training and instruction tuning, by using basic task instructions. The prompt is shown in Figure 1 (a).
- **Role Prompting**: We explore whether assigning a task-related role at the beginning of the prompt contributes to improving zero-shot KE performance, as illustrated in Figure 1 (b).
- **Candidate-based Prompting**: Similar to traditional UKE methods, we provide candidates as context and instruct the model to select the top-k among them (Figure 1 (c)). This strategy evaluates whether LLMs can be utilized as candidate importance estimators.
- **Hybrid Prompting**: We combine the predicted keyphrases obtained through vanilla and candidate-based prompting (both with role prompting applied), instructing the LLMs to make a final keyphrase prediction. This approach investigates the potential for ensemble of these prompting strategies. The example is shown in (d) of Figure 1.

We conduct comprehensive experiments on six widely-used benchmark datasets for KE, primarily focusing on `Llama3-8B-Instruct`, which can run inference on GPUs with 24GB memory, representing a practical scale for real-world applications. Our results show that `Llama3-8B-Instruct`, with vanilla prompting, can outperform the state-of-the-art KE method PromptRank by an average of 9.43%, 7.68%, and 4.82% in F1@5, F1@10, and F1@15, respectively. Furthermore, hybrid prompting improves LLMs' vanilla performance by an average of 3.52%, 1.42%, and 0.68% in F1@5, F1@10, and F1@15, respectively. Through ablation studies, we demonstrate that role prompting consistently improves `Llama3-8B-Instruct`'s zero-shot performance across almost all cases. Additionally, we conduct comprehensive comparative experiments with ChatGPT and several recently released open-source LLMs, demonstrating that Llama3 and Gemma2 can be reasonable choices.

We hope this paper can provide insights for researchers exploring LLMs in KE, as well as practical guidance for model selection in real-world applications.

## 2 Related Work

### 2.1 Unsupervised Keyphrase Extraction

Traditional UKE methods involve extracting noun phrases from a given document to form a candidate set. They estimate the importance of each candidate and rank them to extract the top-k keyphrases. To estimate the importance of candidates, statistics-based methods (Sparck Jones, 1988; Campos et al., 2018) and graph-based methods have been proposed (Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Boudin, 2018). With the advancement of deep learning and pre-trained language models, embedding-based methods have shown strong performance by estimating the semantic relevance between candidates and the input document (Bennani-Smires et al., 2018; Sun et al., 2020; Liang et al., 2021; Zhang et al., 2022; Song et al., 2023). Recently, Kong et al. (2023) proposed PromptRank, a novel approach where the document is input to the encoder of T5 (Raffel et al., 2020), and a prompt-candidate pair template is input to the decoder, estimating the importance of candidates based on the probability that the decoder's prompt generates the candidate.

[1] `https://openai.com/index/chatgpt`

### 2.2 Large language models and Prompting

Since the advent of LLMs (Brown et al., 2020; Ouyang et al., 2022), numerous NLP tasks can be approached through few-shot or zero-shot learning without task-specific training, leading to intensive exploration of prompting techniques. Particularly in the domain of reasoning, various methods have been proposed. To enhance reasoning performance on complex problems, methods such as Chain of Thought (CoT) (Wei et al., 2022), which creates a reasoning chain between the problem and the solution, Zero-CoT (Kojima et al., 2022), which prompts LLMs to generate their reasoning chain using the prompt "Let's think step by step." and Plan and Solve (Wang et al., 2023), which prompts LLMs to devise a plan and solve tasks accordingly, have been proposed.

In the field of Information Extraction (IE), the use of LLMs and prompting has also been explored. Wei et al. (2024) demonstrated that by prompting ChatGPT, a powerful zero-shot IE model could be derived for tasks such as entity relation triple extraction, named entity recognition (NER), and event extraction. Xie et al. (2023) conducted a systematic empirical study on NER using ChatGPT with various prompting techniques.

Following this trend, benchmarking and preliminary studies have been conducted to utilize LLMs for KE and keyphrase generation (Martínez-Cruz et al., 2023; Song et al., 2024a,b). These studies primarily focused on ChatGPT, while the zero-shot KE performance of more recently released LLMs has not yet been deeply investigated. To the best of our knowledge, this study is the first to conduct a comprehensive empirical investigation into the zero-shot KE abilities of various LLMs.

## 3 Method

### 3.1 Vanilla

To find prompts that yield strong zero-shot KE performance, we conduct preliminary experiments on various prompts (See Appendix A). The prompt "Extract keyphrases from the text." is selected as our vanilla prompt, as it performed the best on average. The template for the vanilla prompt is shown in Figure 1 (a).

### 3.2 Role Prompting

Kong et al. (2024) demonstrated that role-play prompting, which assigns specific roles to LLMs, improves their zero-shot reasoning performance. Inspired by this, we assign the role of "Keyphrase extractor" to the LLMs. This role, selected through preliminary experiments (See Appendix B), is implemented by prepending "You are a keyphrase extractor." to the prompt, as shown in Figure 1 (b).

### 3.3 Candidate-based Prompting

Traditional UKE methods compute the importance of each candidate in a set of noun phrases extracted from the input document, rank them, and then extract the top k to predict the final keyphrases. Similar to traditional methods, we investigate whether LLMs can be utilized as ***candidate importance estimators***. We first perform POS tagging using the widely-used StanfordCoreNLP[2] tool and then extract noun phrases using NLTK[3]'s RegexpParser. The extraction is based on a regex pattern `<NN.*|JJ>*<NN.*>` that captures sequences of optional adjectives followed by one or more nouns to form a candidate set. Subsequently, we provide this candidate set along with the document in the prompt, asking the LLMs to extract the top k keyphrases from the candidate set. The template for candidate-based prompting is shown in Figure 1 (c).

### 3.4 Hybrid Prompting

The vanilla prompting relies entirely on the LLM to recognize keyphrases within the text and determine their boundaries independently. In contrast, the candidate-based prompting presents a pre-extracted set of candidates to the LLM, instructing it to select the relatively more important ones, utilizing the model's ability to discriminate between candidates.

To leverage the strengths of both approaches, we propose a hybrid prompting strategy that ensembles two different perspectives of prompting. This approach first concatenates the keyphrases predicted by the vanilla prompt with the top k keyphrases selected through candidate-based prompting (both with role prompting applied) to form the final keyphrase candidates. Then, we instruct the LLMs to predict the final keyphrases from this combined set through another stage of inference. The template for hybrid prompting is depicted in Figure 1 (d).

Our hybrid prompting strategy is designed to address the limitations of each individual method by leveraging their complementary strengths. Thus, it has the potential to capture keyphrases that might

[2] https://stanfordnlp.github.io/CoreNLP
[3] https://github.com/nltk

| Dataset | Domain | $N_{Doc}$ | $L_{Doc}$ | $N_{KP}$ | $N_{Can}$ |
|---|---|---|---|---|---|
| Inspec | Scientific | 500 | 122 | 9.8 | 27.2 |
| SemEval2017 | Scientific | 494 | 170 | 17.3 | 36.8 |
| SemEval2010 | Scientific | 100 | 190 | 15.1 | 36.1 |
| DUC2001 | News | 307 | 725 | 8.1 | 93.0 |
| NUS | Scientific | 211 | 7702 | 11.7 | 92.0 |
| Krapivin | Scientific | 460 | 8545 | 5.7 | 90.5 |

Table 1: Statistics of testing datasets. $N_{Doc}$: number of documents; $L_{Doc}$: average document length; $N_{KP}$: average number of gold keyphrases per document; $N_{Can}$: average number of candidate per document (based on truncated input of 512 words).

be missed by one approach but identified by the other, offering a more comprehensive keyphrase extraction.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets** To comprehensively evaluate the performance of LLMs, we conduct experiments using six keyphrase extraction benchmark datasets widely used in previous studies: Inspec (Hulth, 2003); SemEval2017 (Augenstein et al., 2017); SemEval2010 (Kim et al., 2010); DUC2001 (Wan and Xiao, 2008); NUS (Nguyen and Kan, 2007); Krapivin (Krapivin et al., 2009). The domains and statistical information of each dataset are presented in Table 1. For a fair comparison with baselines, the maximum length of input documents is set to 512.

**Evaluation Metrics** Following previous studies (Liang et al., 2021; Zhang et al., 2022; Kong et al., 2023), we use the F1@K score (K = 5, 10, and 15), a widely used metric in KE. Traditional UKE methods extract the top-k keyphrases based on the importance score of each candidate, but LLMs predict multiple keyphrases as a single text sequence. Therefore, we consider the order of keyphrases predicted by LLMs as their rank for evaluation. Duplicate keyphrases are removed from the predictions, and NLTK's PorterStemmer is applied for word stemming before matching predicted and ground truth keyphrases.

**Baselines** We set the following methods as baselines: the statistical method YAKE (Campos et al., 2020); graph-based methods TextRank (Mihalcea and Tarau, 2004), SingleRank (Wan and Xiao, 2008), TopicRank (Bougouin et al., 2013), MultipartiteRank (Boudin, 2018); embedding-based methods EmbedRank (Bennani-Smires et al., 2018), SIFRank (Sun et al., 2020), MDERank (Zhang et al., 2022); and the method PromptRank (Kong et al., 2023), which is based on the probability that the T5 decoder's prompt generates the candidate.

**LLMs** In this study, we primarily experiment with `Llama-3-8B-Instruct`[4], released by Meta on Hugging Face. Additionally, we conduct experiments on other open-source LLM series, Qwen2 and Gemma2. We also evaluate ChatGPT using the OpenAI API.

**Implementation Details** The experiments conducted in this paper use the source code released by Kong et al. (2023) with PromptRank for data preprocessing and evaluation. To ensure reproducibility, we set `do_sample=False` for open-source LLMs and set `temperature=0` when calling the ChatGPT API. Experiments are conducted on an NVIDIA GeForce RTX 4090 24GB. For large-scale models of 70B parameters and above, we use two NVIDIA H100 80GB GPUs.

### 4.2 Main Result

Table 2 shows the zero-shot KE performance of Llama3-8B-Instruct across six datasets. The results indicate that the simplest prompt, "Extract keyphrases from the text." referred to as vanilla, achieves state-of-the-art (SOTA) performance on most datasets. On average, vanilla outperforms the previous best-performing method, PromptRank, with improvements by approximately 9.43% in F1@5, 7.68% in F1@10, and 4.82% in F1@15. Notably, on the NUS dataset, which contains longer documents, vanilla surpasses PromptRank by 31.67%, 25.68%, and 16.05% in F1@5, F1@10, and F1@15, respectively. Similarly, on the Krapivin dataset, vanilla shows improvements of 33.15%, 23.61%, and 14.36% in F1@5, F1@10, and F1@15, respectively. These results suggest that the prompting-based approach is more effective for longer documents, where the traditional ranking methods struggle with the increased number of candidates. This demonstrates the ability of LLMs to make more refined keyphrase predictions on longer documents. This highlights the efficiency and power of LLMs as zero-shot keyphrase extractors with simple instructions.

[4] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

| F1@K | Method | Dataset | | | | | | AVG |
|---|---|---|---|---|---|---|---|---|
| | | Inspec | SemEval2017 | SemEval2010 | DUC2001 | NUS | Krapivin | |
| 5 | TextRank | 21.58 | 16.43 | 7.42 | 11.02 | 1.80 | 6.04 | 10.72 |
| | SingleRank | 14.88 | 18.23 | 8.69 | 19.14 | 2.98 | 8.12 | 12.01 |
| | TopicRank | 12.20 | 17.10 | 9.93 | 19.97 | 4.54 | 8.94 | 12.11 |
| | MultipartiteRank | 13.41 | 17.39 | 10.13 | 21.70 | 6.17 | 9.29 | 13.02 |
| | YAKE | 8.02 | 11.84 | 6.82 | 11.99 | 7.85 | 8.09 | 9.10 |
| | EmbedRank (BERT) | 28.92 | 20.03 | 10.46 | 8.12 | 3.75 | 4.05 | 12.56 |
| | SIFRank (ELMo) | 29.38 | 22.38 | 11.16 | 24.30 | 3.01 | 1.62 | 15.31 |
| | MDERank (BERT) | 26.17 | 22.81 | 12.95 | 13.05 | 15.24 | 11.78 | 17.00 |
| | PromptRank (T5) | 31.73 | **27.14** | 17.24 | 27.39 | 17.24 | 16.11 | 22.81 |
| | Vanilla (Llama3-8B-Instruct) | 36.75 | 24.21 | 17.04 | 27.60 | 22.70 | **21.45** | 24.96 |
| | Hybrid (Llama3-8B-Instruct) | **36.81** | 25.12 | **18.44** | **29.98** | **23.27** | 21.41 | **25.84** |
| 10 | TextRank | 27.53 | 25.83 | 11.27 | 17.45 | 3.02 | 9.43 | 15.76 |
| | SingleRank | 21.50 | 27.73 | 12.94 | 23.86 | 4.51 | 10.53 | 16.85 |
| | TopicRank | 17.24 | 22.62 | 12.52 | 21.73 | 7.93 | 9.01 | 15.18 |
| | MultipartiteRank | 18.18 | 23.73 | 12.91 | 24.10 | 8.57 | 9.35 | 16.14 |
| | YAKE | 11.47 | 18.14 | 11.01 | 14.18 | 11.05 | 9.35 | 12.53 |
| | EmbedRank (BERT) | 38.55 | 31.01 | 16.35 | 11.62 | 6.34 | 6.60 | 18.41 |
| | SIFRank (ELMo) | 39.12 | 32.60 | 16.03 | 27.60 | 5.34 | 2.52 | 20.54 |
| | MDERank (BERT) | 33.81 | 32.51 | 17.07 | 17.31 | 18.33 | 12.93 | 21.99 |
| | PromptRank (T5) | 37.88 | **37.76** | 20.66 | **31.59** | 20.13 | 16.71 | 27.46 |
| | Vanilla (Llama3-8B-Instruct) | **46.42** | 34.95 | **22.46** | 27.74 | **25.30** | 20.58 | 29.57 |
| | Hybrid (Llama3-8B-Instruct) | 45.80 | 36.08 | 22.45 | 29.55 | 24.99 | **21.09** | **29.99** |
| 15 | TextRank | 27.62 | 30.50 | 13.47 | 18.84 | 3.53 | 9.95 | 17.32 |
| | SingleRank | 24.13 | 31.73 | 14.40 | 23.43 | 4.92 | 10.42 | 18.17 |
| | TopicRank | 19.33 | 24.87 | 12.26 | 20.97 | 9.37 | 8.30 | 15.85 |
| | MultipartiteRank | 20.52 | 26.87 | 13.24 | 23.62 | 10.82 | 9.16 | 17.37 |
| | YAKE | 13.65 | 20.55 | 12.55 | 14.28 | 13.09 | 9.12 | 13.87 |
| | EmbedRank (BERT) | 39.77 | 36.72 | 19.35 | 13.58 | 8.11 | 7.84 | 20.90 |
| | SIFRank (ELMo) | 39.82 | 37.25 | 18.42 | 27.96 | 5.86 | 3.00 | 22.05 |
| | MDERank (BERT) | 36.17 | 37.18 | 20.09 | 19.13 | 17.95 | 12.58 | 23.85 |
| | PromptRank (T5) | 38.17 | **41.57** | 21.35 | **31.01** | 20.12 | 16.02 | 28.04 |
| | Vanilla (Llama3-8B-Instruct) | **48.16** | 38.56 | **22.89** | 25.09 | 23.35 | 18.32 | 29.39 |
| | Hybrid (Llama3-8B-Instruct) | 45.17 | 40.23 | 22.87 | 27.21 | **23.70** | **18.35** | **29.59** |

Table 2: F1@K performance of zero-shot keyphrase extraction with Llama3-8B-Instruct on six datasets, for K ∈ {5, 10, 15}. The best performance is **bold** and the second-best is underlined.

In Table 2, hybrid refers to the experimental results where LLMs re-predict the final keyphrases from a set that combines the keyphrases predicted using the vanilla prompt with the top-5 keyphrases predicted using candidate-based prompting. For both stages, the prompts begin with a role assignment. The results show that the hybrid method improves upon the vanilla performance in most cases, with average improvements of 3.52%, 1.42%, and 0.68% in F1@5, F1@10, and F1@15, respectively. The impact of each component is discussed in detail in the ablation study in the following section 4.3.

### 4.3 Ablation Study

#### 4.3.1 Effects of Role Prompting

Table 3 shows the performance when applying role prompting to each proposed prompting strategy compared to when it is not applied. The results indicate that role prompting enhances performance not only for the vanilla prompt but also for candidate-based prompting and the combined Hybrid method. Notably, in some cases, such as SemEval's F1@10 and F1@15, the simpler role prompting outperforms the more complex Hybrid method. This suggests that role prompting, despite merely adding a single sentence to the prompt, can play a critical role in improving zero-shot performance.

To further analyze the impact of the assigned role on LLM performance, we conducted experiments applying various role prompts to the vanilla prompt, as discussed in B. The results show that assigning the role of 'keyphrase extractor' consistently improves the zero-shot performance of the vanilla prompt. Conversely, assigning roles irrelevant to the task leads to decreased performance

| F1@K | Method | Dataset | | | | | | AVG |
|---|---|---|---|---|---|---|---|---|
| | | Inspec | SemEval2017 | SemEval2010 | DUC2001 | NUS | Krapivin | |
| 5 | PromptRank (T5) | 31.73 | **27.14** | 17.24 | 27.39 | 17.24 | 16.11 | 22.81 |
| | Vanilla w/o Role | 36.75 | 24.21 | 17.04 | 27.60 | 22.70 | 21.45 | 24.96 |
| | Vanilla w/ Role | 36.66 | 24.90 | 17.24 | 28.16 | 22.47 | 21.17 | 25.10 |
| | Candidate w/o Role | 31.89 | 23.46 | 17.74 | 29.58 | 18.09 | 17.93 | 23.12 |
| | Candidate w/ Role | 33.57 | 23.84 | **18.63** | **30.43** | 20.53 | 19.39 | 24.40 |
| | Hybrid w/o Role | **37.18** | 24.52 | 18.04 | 28.88 | 23.09 | **21.62** | 25.56 |
| | Hybrid w/ Role | 36.81 | 25.12 | 18.44 | 29.98 | **23.27** | 21.41 | **25.84** |
| 10 | PromptRank (T5) | 37.88 | **37.76** | 20.66 | 31.59 | 20.13 | 16.71 | 27.46 |
| | Vanilla w/o Role | **46.42** | 34.95 | 22.46 | 27.74 | **25.30** | 20.58 | 29.58 |
| | Vanilla w/ Role | 46.24 | 35.75 | **22.92** | 28.65 | 25.04 | 20.86 | 29.91 |
| | Candidate w/o Role | 40.57 | 33.36 | 21.62 | 31.10 | 21.40 | 18.37 | 27.74 |
| | Candidate w/ Role | 42.18 | 33.06 | 21.70 | **32.31** | 23.19 | 19.39 | 28.64 |
| | Hybrid w/o Role | 46.27 | 35.40 | 21.94 | 28.73 | 24.90 | 20.66 | 29.65 |
| | Hybrid w/ Role | 45.80 | 36.08 | 22.45 | 29.55 | 24.99 | **21.09** | **29.99** |
| 15 | PromptRank (T5) | 38.17 | **41.57** | 21.35 | **31.01** | 20.12 | 16.02 | 28.04 |
| | Vanilla w/o Role | **48.16** | 38.56 | 22.89 | 25.09 | 23.35 | 18.32 | 29.40 |
| | Vanilla w/ Role | 48.04 | 39.68 | **23.21** | 25.79 | 23.53 | 18.27 | **29.75** |
| | Candidate w/o Role | 41.40 | 37.60 | 21.68 | 28.56 | 20.76 | 17.57 | 27.93 |
| | Candidate w/ Role | 42.30 | 37.53 | 22.95 | 29.76 | 22.61 | **18.50** | 28.94 |
| | Hybrid w/o Role | 45.85 | 39.50 | 23.10 | 26.83 | 23.48 | 18.34 | 29.52 |
| | Hybrid w/ Role | 45.17 | 40.23 | 22.87 | 27.21 | **23.70** | 18.35 | 29.59 |

Table 3: Ablation study results on six datasets with Llama3-8B-Instruct. "Vanilla" refers to the predictions from (a) in 1. "Candidate" refers to the prompt format shown in (c) of 1, but with the instruction to extract the top 15 instead of top 5. "Hybrid" refers to the combination of the output from (a) in 1 and the top-5 outputs obtained from (c) to make the final prediction through (d). "w/ Role" refers to adding the sentence "You are a keyphrase extractor." at the beginning of the prompt, whereas "w/o Role" refers to the case where this sentence was not included. The best performance is **bold**.

across all metrics compared to the vanilla prompt. This demonstrates that simply assigning any role does not enhance performance; rather, the role must be relevant and helpful for the LLM to understand the task to achieve performance improvements.

#### 4.3.2 Performance of Candidate-based Prompting

We evaluated the performance of candidate-based prompting alone to determine if LLMs could effectively select keyphrases from a candidate set. To assess performance across F1@5, F1@10, and F1@15, we modified the prompt in Figure 1 (c) by changing "top 5" to "top 15". As shown in Table 3, candidate-based prompting generally performs lower than the vanilla prompt on average. However, when a role is assigned, the average performance surpasses that of PromptRank, the best among traditional KE methods. This indicates that while directly extracting keyphrases might be more effective, LLMs are still capable of accurately identifying relatively important keyphrases from a candidate set more effectively than traditional methods.

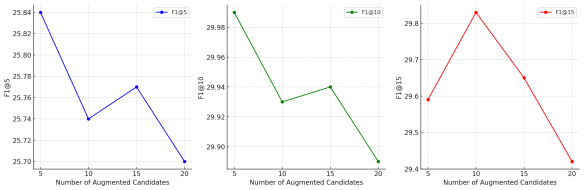


Figure 2: Average F1@5, F1@10, and F1@15 performance across 6 datasets for hybrid prompting with increasing number of augmented candidates.

#### 4.3.3 Effects of Augmented Candidates

In Table 3, Hybrid refers to the results where the final keyphrases are re-predicted from a set that combines the keyphrases predicted by the vanilla prompt with five additional keyphrases selected through candidate-based prompting. The results show that Hybrid generally improves performance over the vanilla prompt, both with and without role assignment. This indicates that the candidates selected by LLMs can serve as valuable information for predicting the final keyphrases.

We also evaluated the impact of the number of keyphrases (k) augmented on the final Hybrid pre-

dictions by experimenting with k values of 5, 10, 15, and 20. Figure 2 shows the performance of F1@5, F1@10, and F1@15 as the number of augmented candidates increases. The highest performance for F1@5 and F1@10 is observed when augmenting with five candidates, while performance decreases when augmenting with 15 or more candidates. This suggests that simply increasing the number of augmented candidates does not contribute to performance improvements; rather, augmenting with a few selected candidates by the LLMs is more effective.

### 4.4 Experiments with Larger and Different LLMs

To investigate the KE performance of various LLMs and evaluate the generalizability of our approach, we conducted experiments on different series of LLMs recently released, specifically GPT-4o, Gemma2, and Qwen2. We also conducted experiments across various model scales to investigate the impact of model size on performance. We discovered that GPT-4o tends to generate longer phrases when the term 'keyphrase' is used in prompts. To address this, we experimented with using 'keyword' instead of 'keyphrase' for GPT-4o, denoted as GPT-4o* in Table 4.

Table 4 presents the results of our comprehensive evaluation, showing the average performance across all datasets. Detailed results for each dataset can be found in Appendix C. The hybrid prompting approach demonstrates consistent improvement over vanilla prompting across most models, with the exception of Qwen2-7B-Instruct and GPT-4o. Further analysis revealed that while hybrid prompting itself enhances performance for Qwen2-7B-Instruct, the application of role-prompting leads to an overall performance decrease. Interestingly, for GPT-4o*, where we used 'keyword' instead of 'keyphrase', the hybrid approach does improve performance, further emphasizing the generality of our method.

Our results reveal a general trend of performance improvement with increased model size for Llama3, Gemma2, and Qwen2, especially when using the hybrid prompting strategy. However, for some models, we observe that larger-scale models exhibit lower performance in vanilla prompting. We speculate that larger models may be more sensitive to specific vocabulary in prompts, potentially leading to variations in phrase length and structure. We discuss more detail in Appendix C.

| Model | Method | F1@K | | |
|---|---|---|---|---|
| | | 5 | 10 | 15 |
| T5-base | PromptRank | 22.81 | 27.46 | 28.04 |
| Llama3-8B-Instruct | Vanilla | 24.96 | 29.57 | 29.39 |
| | Hybrid | <u>25.84</u> | 29.99 | 29.59 |
| Llama3-70B-Instruct | Vanilla | 24.30 | 28.44 | 28.45 |
| | Hybrid | **26.22** | **30.54** | **30.85** |
| Gemma2-9b-it | Vanilla | 24.29 | 28.07 | 28.07 |
| | Hybrid | 25.28 | 29.15 | 29.64 |
| Gemma2-27b-it | Vanilla | 24.95 | 29.12 | 29.12 |
| | Hybrid | 25.48 | <u>30.18</u> | <u>30.84</u> |
| Qwen2-7B-Instruct | Vanilla | 19.29 | 23.93 | 24.99 |
| | Hybrid | 18.04 | 22.34 | 23.44 |
| Qwen2-72B-Instruct | Vanilla | 21.09 | 26.03 | 27.06 |
| | Hybrid | 21.81 | 26.66 | 27.96 |
| GPT-4o-mini | Vanilla | 22.28 | 27.68 | 28.44 |
| | Hybrid | 22.83 | 28.09 | 28.72 |
| GPT-4o | Vanilla | 19.78 | 25.27 | 27.06 |
| | Hybrid | 19.65 | 25.01 | 26.89 |
| GPT-4o* | Vanilla | 21.52 | 27.58 | 29.46 |
| | Hybrid | 22.14 | 28.43 | 30.72 |

Table 4: Performance comparison of various LLMs on zero-shot keyphrase extraction. Results are shown for different model sizes and two prompting methods: vanilla and hybrid. F1@K scores are reported for K=5, 10, and 15. The best scores are in **bold**, while the second-best are <u>underlined</u>. * denotes experiments where 'keyword' was used instead of 'keyphrase' in prompts.

In summary, among the recently released LLMs, Llama3 and Gemma2 demonstrated the strongest zero-shot KE performance. Our experiments show that hybrid prompting can universally improve the inherent vanilla performance of various LLMs, not just Llama3. These results demonstrate the broad applicability and effectiveness of our proposed method in enhancing zero-shot KE across various LLM series and scales.

## 5 Analysis

To understand why hybrid prompting improves zero-shot KE performance, we performed a detailed analysis. Our analysis reveals that hybrid prompting effectively combines the strengths of vanilla and candidate-based prompting.

Figure 3 illustrates the hybrid prediction process using Llama3-8B-Instruct for a document from the DUC2001 dataset in the News domain. In stage I (vanilla prompting), 17 keyphrases are predicted, matching 3 gold keyphrases. Stage II (candidate-based prompting) identifies 2 additional gold keyphrases missed in stage I. Notably, in stage III (hybrid prompting), these newly identified gold keyphrases are ranked among the top 5, demonstrating the model's ability to reassess and refine keyphrase importance.

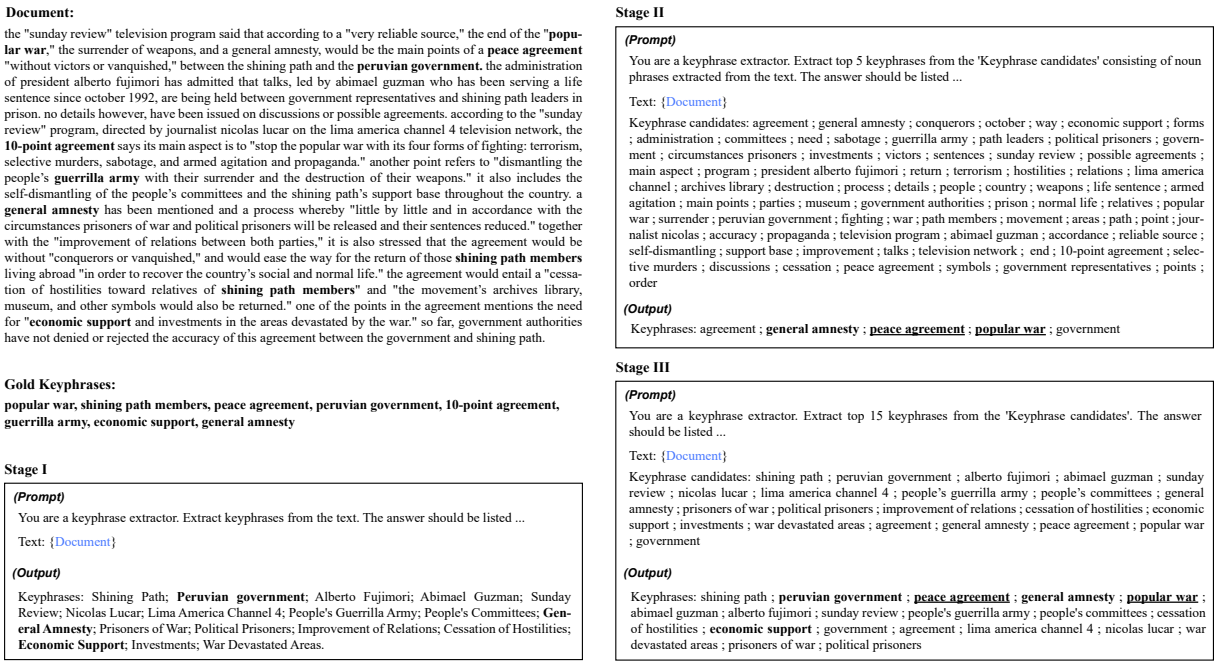


Figure 3: Example of hybrid prompting for keyphrase extraction using Llama3-8B-Instruct on a document from the DUC2001 dataset.

We examined the source of keyphrases in hybrid prompting predictions, with results presented in Table 5. Keyphrases predicted by both vanilla and candidate-based prompting methods (V∩C) made up about 20% of final predictions and showed the highest precision, suggesting that agreement between methods enables more refined keyphrase predictions. Vanilla-only predictions (V-C) formed the largest portion (44-72%) of final keyphrases with high precision, indicating that the model's inherent vanilla performance plays a comparatively larger role than the candidate-based approach. Despite being limited to just 5 candidates, candidate-based prompting-only predictions (C-V) accounted for 4-10% of final keyphrases with high precision. Hybrid prompting also predicted new keyphrases (∼(V∪C)), ranging from 1.9% to 21.4% across datasets, some of which were correct. These results demonstrate that hybrid prompting effectively combines different perspectives, leading to more comprehensive and accurate keyphrase extraction. This method successfully utilizes the strengths of individual approaches while complementing each other's limitations, allowing it to predict keyphrases that individual methods might miss.

| Dataset | V∩C (prec.) | V-C (prec.) | C-V (prec.) | ∼(V∪C) (prec.) |
|---|---|---|---|---|
| Insepc | 24.1 (65.99) | 44.4 (40.54) | 10.1 (28.65) | 21.4 (11.44) |
| SemEval2017 | 21.7 (62.35) | 56.2 (42.82) | 8.2 (37.10) | 13.9 (20.36) |
| SemEval2010 | 23.0 (41.59) | 54.3 (20.25) | 8.3 (20.33) | 14.4 (6.13) |
| DUC2001 | 21.3 (43.33) | 67.7 (9.46) | 7.4 (23.84) | 3.6 (11.37) |
| NUS | 22.0 (40.27) | 71.6 (11.08) | 4.3 (16.37) | 2.0 (7.41) |
| Krapivin | 22.0 (23.87) | 71.6 (6.82) | 4.5 (6.65) | 1.9 (7.74) |

Table 5: Distribution (%) and precision (in parentheses) of keyphrase predictions in hybrid prompting. V∩C: predicted by both vanilla and candidate-based prompting; V-C: predicted only by vanilla prompting; C-V: predicted only by candidate-based prompting; ∼(V∪C): newly predicted by hybrid prompting.

## 6 Error Analysis

To gain deeper insights into how LLMs fail at KE, we conducted a systematic error analysis by categorizing prediction errors into five types:

- **Over**: The predicted keyphrase extends beyond the gold keyphrase's boundaries while containing it fully.
- **Partial**: The predicted keyphrase captures only a fragment of the gold keyphrase.
- **Misordered**: The predicted keyphrase consists of exactly the same components as the gold keyphrase but in a different sequence.
- **Intersection**: The predicted keyphrase partially overlaps with the gold keyphrase.
- **Unrecognized**: The predicted keyphrase has no lexical overlap with the gold keyphrase.

Table 6 presents the error analysis results of Llama3-8B-Instruct with hybrid prompting across

| Error Type | Percentage | Example |
|---|---|---|
| Over | 13.96% | **Gold Keyphrase**: computer applications<br>**Predicted Keyphrase**: computer applications in power, ... |
| Partial | 14.68% | **Gold Keyphrase**: legislative term limits<br>**Predicted Keyphrase**: term limits, ... |
| Misorderd | 0.76% | **Gold Keyphrase**: pulse sequence decoupling<br>**Predicted Keyphrase**: decoupling pulse sequences, ... |
| Intersection=1 | 33.05% | **Gold Keyphrase**: vulnerable fire zones<br>**Predicted Keyphrase**: fire resistant species, ... |
| Intersection=2 | 3.96% | **Gold Keyphrase**: time critical data<br>**Predicted Keyphrase**: time sensitive data, ... |
| Intersection>=3 | 0.70% | **Gold Keyphrase**: multi agent distributed system<br>**Predicted Keyphrase**: multi agent system, ... |
| Unrecognized | 32.88% | **Gold Keyphrase**: exchange hamiltonian<br>**Predicted Keyphrase**: solid state quantum computing, decoherence, decoupling solution, ... |

Table 6: Error analysis results for Llama3-8B-Instruct with hybrid prompting across six datasets. Percentages indicate the proportion of each error type in the total prediction errors. Examples show gold keyphrases and their corresponding predicted keyphrases, with matching words highlighted in blue.

all six datasets. Approximately 28.6% of all errors (Over: 13.96%, Partial: 14.68%) stem from inaccurate boundary detection of gold keyphrases. This suggests that LLMs are actually predicting keyphrases quite close to the correct answers, but these predictions are considered incorrect under the widely used exact match-based evaluation. This finding indicates that semantic-based evaluation methods, such as the recently released KPEval (Wu et al., 2024), might be more appropriate for assessing KE performance with LLMs.

Word order errors (Misordered) account for only 0.76% of all errors. For Intersection errors, cases sharing only one word with the gold keyphrase (Intersection=1) comprise 33.05% of errors, with the percentage decreasing sharply as the number of shared words increases (Intersection=2: 3.96%, Intersection>=3: 0.70%). Finally, Unrecognized errors, where predictions share no words with gold keyphrases, account for 32.88% of all errors, indicating that LLMs still face significant challenges in recognizing keyphrases.

## 7 Conclusion

This study empirically demonstrates that a prompting-based approach utilizing LLMs can effectively supersede traditional KE methods. We show that even simple vanilla prompting, which provides basic task instructions, significantly outperforms PromptRank, the previous state-of-the-art method. Furthermore, we propose candidate-based prompting, which leverages LLMs as candidate importance estimators. By combining this with vanilla prompting in our novel hybrid prompting strategy, we achieve further improvements in LLMs' zero-shot KE performance. We also demonstrate that simply adding a task-related role at the beginning of the prompt can enhance zero-shot KE performance.

Through comprehensive experiments across various LLM series and scales, we identify Llama3 and Gemma2 as the most effective among recently released LLMs for KE tasks. Notably, we show that even models with approximately 8 billion parameters, which can be run on GPUs with 24GB of memory, can achieve higher performance than traditional KE methods. This finding underscores the practical applicability of our approach in real-world scenarios.

## 8 Limitations

As pointed out by Golchin and Surdeanu (2024), data contamination, where test data from downstream tasks is present in the training data of LLMs, impacts the evaluation of LLMs. The datasets used in our experiments might overlap with the training corpora of the LLMs, potentially influencing the assessment of the models' inherent keyphrase extraction capabilities in zero-shot settings.

Moreover, we employ simple prompts, and it's possible that more detailed instructions could lead to different results. The choices made in designing our prompts were not entirely data-driven and involved some degree of arbitrariness, which may have impacted performance.

Finally, LLMs demand significantly more computational resources compared to traditional methods, making them more expensive to run. This

higher computational cost can limit their practicality in certain applications, especially where resources are constrained.

## Acknowledgments

This work was supported by the IITP(Institute of Information & Coummunications Technology Planning & Evaluation)-ICAN(ICT Challenge and Advanced Network of HRD)(IITP-2025-RS-2023-00260175) grant funded by the Korea government(Ministry of Science and ICT) and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(RS-2024-00352711).

## References

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555, Vancouver, Canada. Association for Computational Linguistics.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 221–229, Brussels, Belgium. Association for Computational Linguistics.

Florian Boudin. 2018. Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672, New Orleans, Louisiana. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2018. Yake! collection-independent automatic keyword extractor. In *Advances in Information Retrieval*, pages 806–810, Cham. Springer International Publishing.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Shahriar Golchin and Mihai Surdeanu. 2024. Time travel in LLMs: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden. Association for Computational Linguistics.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xiaoyan Bai. 2023. PromptRank: Unsupervised keyphrase extraction using prompt. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9788–9801, Toronto, Canada. Association for Computational Linguistics.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, Xin Zhou, Enzhi Wang, and Xiaohang Dong. 2024. Better zero-shot reasoning with role-play prompting. *Preprint*, arXiv:2308.07702.

Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. Large dataset for keyphrases extraction.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Unsupervised keyphrase extraction by jointly modeling local and global context. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 155–164, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Roberto Martínez-Cruz, Alvaro Lopez Lopez, and José Portela. 2023. Chatgpt vs state-of-the-art models: A benchmarking study in keyphrase generation task.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, pages 317–326, Berlin, Heidelberg. Springer Berlin Heidelberg.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Mingyang Song, Xuelian Geng, Songfang Yao, Shilong Lu, Yi Feng, and Liping Jing. 2024a. Large language models as zero-shot keyphrase extractors: A preliminary empirical study. *Preprint*, arXiv:2312.15156.

Mingyang Song, Xuelian Geng, Songfang Yao, Shilong Lu, Yi Feng, and Liping Jing. 2024b. Large language models as zero-shot keyphrase extractors: A preliminary empirical study. *Preprint*, arXiv:2312.15156.

Mingyang Song, Huafeng Liu, and Liping Jing. 2023. HyperRank: Hyperbolic ranking model for unsupervised keyphrase extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16070–16080, Singapore. Association for Computational Linguistics.

Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Yi Sun, Hangping Qiu, Yu Zheng, Zhongwei Wang, and Chaoran Zhang. 2020. Sifrank: A new baseline for unsupervised keyphrase extraction based on pre-trained language model. *IEEE Access*, 8:10896–10906.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, page 855–860. AAAI Press.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2024. Chatie: Zero-shot information extraction via chatting with chatgpt. *Preprint*, arXiv:2302.10205.

Di Wu, Da Yin, and Kai-Wei Chang. 2024. KPEval: Towards fine-grained semantic-based keyphrase evaluation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1959–1981, Bangkok, Thailand. Association for Computational Linguistics.

Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. 2023. Empirical study of zero-shot NER with ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7935–7956, Singapore. Association for Computational Linguistics.

Linhan Zhang, Qian Chen, Wen Wang, Chong Deng, ShiLiang Zhang, Bing Li, Wei Wang, and Xin Cao. 2022. MDERank: A masked document embedding rank approach for unsupervised keyphrase extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 396–409, Dublin, Ireland. Association for Computational Linguistics.

## A Exploring Robust Vanilla Prompt

To explore a robust vanilla prompt that can leverage the strong zero-shot KE capabilities of LLMs, we conducted experiments by combining various words. The results are shown in Table 8. We found that prompts instructing the extraction of 'keyphrase' generally performed better than those instructing the extraction of 'keyword'. Additionally, using 'key phrase' with a space between 'key' and 'phrase' resulted in decreased performance, suggesting that the noun 'keyphrase' itself may be a crucial trigger token. When referring to the input document, the term 'text' performed best on average compared to 'input text' and 'document', though the performance difference was minimal. Among imperative, polite requests with 'Please', and interrogative forms, the imperative form slightly outperformed the others. Moreover, adding the modifier 'the most important' before 'keyphrase' significantly improved F1@5 performance but tended to reduce F1@10 and F1@15 scores. These findings highlight that the zero-shot performance of LLMs is highly sensitive to the choice of words and format, emphasizing the importance of optimizing prompt selection.

## B Impact of Role

To understand the impact of the role assigned to LLMs, we experimented with the vanilla prompt by assigning roles unrelated to the task. As shown in Table 9, the performance decreased when unrelated roles were assigned compared to when no role was assigned. Conversely, assigning the task-related role 'keyphrase extractor' resulted in better performance overall, especially with the simplest form, "You are a keyphrase extractor." outperforming more elaborate variations. Additionally, we found that the role of 'information extractor' also improved performance. This demonstrates that providing a clear and relevant role related to the task can contribute to improved zero-shot performance.

Is the 'keyphrase extractor' role effective across all LLMs? Table 10 shows the experimental results of role prompting in vanilla prompts for various LLMs. Role prompting improves the performance of vanilla prompts without assigned roles in all LLMs except Qwen2. Moreover, role prompting is effective even in large-scale models. In the DUC2001 dataset, which is in the news domain, role prompting shows a more significant performance improvement compared to other datasets in the scientific domain. However, we observe that Qwen2 tends to show decreased performance when role prompting is applied. We speculate that the effectiveness of role prompting may stem from the prompts used in the training of LLMs, particularly the system prompts. When a prompt format different from that used in the prompts during training is input, it might lead to performance degradation.

## C Detailed Results of Various LLMs

In addition to Llama-3-8B-Instruct, we conducted experiments on Llama-3-70B-Instruct[5], Gemma2-9b-it[6], Gemma2-27b-it[7], Qwen2-7B-Instruct[8], Qwen2-72B-Instruct[9], GPT-4o[10], and GPT-4o-mini[11].

We observed that when using the vanilla prompt, GPT-4o's performance was lower than that of GPT-4o-mini. Our analysis suggests this may be due to the vocabulary used in the prompt. As shown in Table 7, GPT-4o generates a lower proportion of one- and two-word phrases, but a higher proportion of keyphrases with three or more words compared to GPT-4o-mini. Similarly, we found that larger-scale Llama3 models generate longer keyphrases at a higher rate. In Table 7, the asterisk (*) denotes results obtained when replacing the term 'keyphrase' with 'keyword' in the prompt, which led to the generation of more short keyphrases. This demonstrates that larger LLMs are highly sensitive to the specific words used in prompts. In such cases, using 'keyword' instead of 'keyphrase' may yield better performance for larger models, as 'keyphrase' might induce the generation of excessively long keyphrases.

Table 11 presents the vanilla and hybrid performance of various LLMs. The hybrid prompting strategy improves upon the vanilla performance for all LLMs except Qwen2 and GPT-4o. As previously mentioned, the performance decrease in Qwen2 may be attributed to the detrimental effect of role prompting, while for GPT-4o, the hybrid approach does improve performance when 'keyword' is used instead of 'keyphrase' (denoted as

[5] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct

[6] https://huggingface.co/google/gemma-2-9b-it

[7] https://huggingface.co/google/gemma-2-27b-it

[8] https://huggingface.co/Qwen/Qwen2-7B-Instruct

[9] https://huggingface.co/Qwen/Qwen2-72B-Instruct

[10] gpt-4o-2024-05-13

[11] gpt-4o-mini-2024-07-18

| Model | Keyphrase Distribution | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | >=5 |
| Llama3-8B-Instruct | 17.2 | 56.9 | 19.0 | 4.9 | 2.1 |
| Llama3-70B-Instruct | 7.3 | 56.9 | 25.1 | 7.5 | 3.2 |
| Llama3-70B-Instruct* | 26.2 | 56.4 | 14.6 | 2.3 | 0.6 |
| GPT-4o-mini | 9.6 | 55.6 | 24.4 | 7.6 | 2.7 |
| GPT-4o | 8.6 | 48.5 | 25.8 | 11.0 | 6.1 |
| GPT-4o* | 23.8 | 53.2 | 18.2 | 3.7 | 1.0 |
| Ground truth | 20.8 | 47.4 | 18.8 | 6.2 | 6.9 |

Table 7: Distribution of keyphrase lengths (in words) for various LLM configurations. * denotes results when 'keyphrase' is replaced with 'keyword' in the prompt. The ground truth distribution is included for comparison.

GPT-4o*). These findings demonstrate the efficacy of the hybrid prompting strategy across multiple LLMs.

Comparing the baseline KE performance of various LLM series using the vanilla approach, we find that Llama3 and Gemma2 exhibit superior KE capabilities overall, with even their 7B-8B scale models significantly outperforming traditional KE methods. In contrast, models like Qwen2 and GPT-4o show performance comparable to or lower than existing methods such as PromptRank, even with the improved performance achieved through hybrid prompting. This indicates that while the proposed prompting strategies can enhance the apparent KE performance of LLMs, the intrinsic KE capabilities of the models play a substantial role in determining overall performance.

## D Performance Variation by Temperature

To investigate the effect of temperature values on zero-shot KE performance, we evaluated the vanilla performance changes of Llama3-8B-Instruct by varying the temperature from 0.0 to 1.5 in increments of 0.1. Since setting the temperature above 0 results in different outputs for each inference, we conducted a total of three experiments and reported the average performance along with the standard deviation. As shown in Figure 4, we observed that performance consistently decreases in F1@5, F1@10, and F1@15 as the temperature increases. In all cases, the highest performance was achieved when the temperature was set to 0.0. This indicates that applying greedy decoding rather than random sampling decoding may be more advantageous for zero-shot keyphrase extraction using LLMs.

| No. | Prompt | F1@K | | |
|---|---|---|---|---|
| | | 5 | 10 | 15 |
| 1 | Extract **key words** from the input text. The answer should be listed after 'Key words: ' and separated by semicolons (;). 'Key words: key word 1 ; key word 2 ; ... ; key word N'<br><br>Input text: {document} | 22.39 | 27.31 | 27.45 |
| 2 | Extract **keywords** from the input text. The answer should be listed after 'Keywords: ' and separated by semicolons (;). 'Keywords: keyword 1 ; keyword 2 ; ... ; keyword N'<br><br>Input text: {document} | 22.78 | 27.41 | 27.89 |
| 3 | Extract **key phrases** from the input text. The answer should be listed after 'Key phrases: ' and separated by semicolons (;). 'Key phrases: key phrase 1 ; key phrase 2 ; ... ; key phrase N'<br><br>Input text: {document} | 21.91 | 26.48 | 26.73 |
| 4 | Extract **keyphrases** from the **input text**. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Input text: {document} | 24.85 | 29.43 | 29.30 |
| 5 | Extract keyphrases from the **document**. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Document: {document} | 24.95 | 29.19 | 29.10 |
| 6 | Extract **the** keyphrases from the **text**. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Text: {document} | 24.84 | 29.30 | 29.26 |
| 7 | Extract keyphrases from the text. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Text: {document} | 24.96 | **29.57** | **29.39** |
| 8 | **Please** extract keyphrases from the text. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Text: {document} | 24.84 | 29.30 | 29.16 |
| 9 | **What are** the keyphrases of the text? The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Text: {document} | 25.00 | 29.02 | 28.67 |
| 10 | Extract the **most important** keyphrases from the text. The answer should be listed after 'Keyphrases: ' and separated by semicolons (;). 'Keyphrases: keyphrase 1 ; keyphrase 2 ; ... ; keyphrase N'<br><br>Text: {document} | **25.22** | 28.71 | 28.35 |

Table 8: Average F1@K performance (K=5, 10, and 15) on six datasets for ten different vanilla prompts.

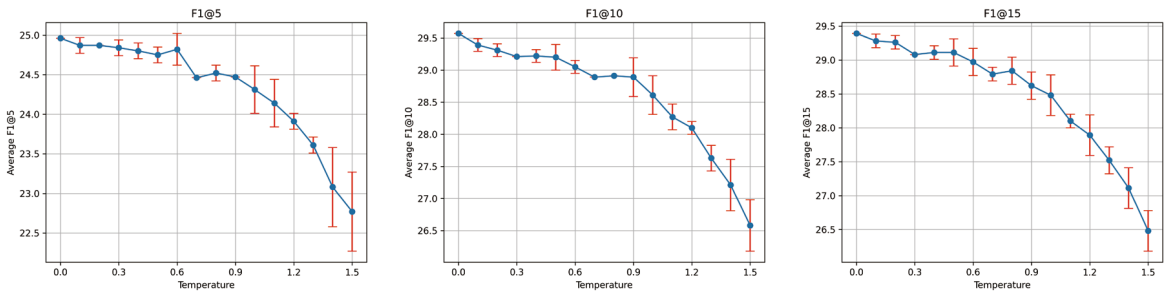


Figure 4: Effect of temperature on F1@K performance. The graphs show F1@5, F1@10, and F1@15 scores of Llama3-8B-Instruct across temperature values from 0.0 to 1.5. Each point represents the average performance over six datasets and three runs, with red error bars indicating the standard deviation.

| Role | Category | F1@K | | |
|---|---|---|---|---|
| | | 5 | 10 | 15 |
| None | vanilla | 24.96 | 29.57 | 29.39 |
| You are a helpful AI assistant. | instructive | 24.49 | 29.07 | 29.15 |
| You are a keyphrase generator. | | 24.44 | 29.17 | 29.30 |
| You are a text summarizer. | | 25.01 | 29.43 | 29.22 |
| You are a information extractor. | | **25.12** | 29.56 | 29.49 |
| **You are a keyphrase extractor.** | | 25.10 | **29.91** | **29.75** |
| You are an excellent keyphrase extractor. | | 24.85 | 29.72 | 29.68 |
| You are a high performance keyphrase extractor. | | 24.80 | 29.73 | 29.66 |
| You are a State-of-the-art (SOTA) keyphrase extractor. | | 25.01 | 29.80 | 29.73 |
| You are a singer. | misleading | 24.67 | 29.14 | 29.18 |
| You are a soccer player. | | 24.40 | 29.14 | 29.06 |
| You are Batman. | | 24.63 | 28.87 | 28.92 |
| You are a joke generator | | 24.17 | 29.11 | 29.30 |

Table 9: Experimental results based on assigning different roles to the vanilla prompt, showing the average performance across six datasets in terms of F1@K (K = 5, 10 and 15).

| F1@K | Model | Method | Dataset | | | | | | AVG |
|---|---|---|---|---|---|---|---|---|---|
| | | | Inspec | SemEval2017 | SemEval2010 | DUC2001 | NUS | Krapivin | |
| 5 | T5-base | PromptRank | 31.73 | **27.14** | **17.24** | 27.39 | 17.24 | 16.11 | 22.81 |
| | Llama3-8B-Instruct | Vanilla w/o Role | 36.75 | 24.21 | 17.04 | 27.60 | 22.70 | 21.45 | 24.96 |
| | | Vanilla w/ Role | 36.66 | 24.90 | **17.24** | 28.16 | 22.47 | 21.17 | 25.10 |
| | Llama3-70B-Instruct | Vanilla w/o Role | 36.93 | 23.67 | 16.64 | 28.96 | 19.68 | 19.91 | 24.30 |
| | | Vanilla w/ Role | **37.29** | 24.30 | 16.94 | **31.44** | 20.31 | 20.64 | 25.15 |
| | Gemma2-9b-it | Vanilla w/o Role | 35.14 | 22.10 | 16.84 | 25.95 | 23.27 | 22.43 | 24.29 |
| | | Vanilla w/ Role | 35.13 | 22.68 | **17.24** | 27.64 | **23.44** | **23.08** | 24.87 |
| | Gemma2-27b-it | Vanilla w/o Role | 36.30 | 23.21 | 16.94 | 28.94 | 22.58 | 21.75 | 24.95 |
| | | Vanilla w/ Role | 36.58 | 23.54 | **17.24** | 30.34 | 23.21 | 21.18 | **25.35** |
| | Qwen2-7B-Instruct | Vanilla w/o Role | 31.41 | 19.61 | 12.46 | 23.08 | 14.45 | 14.75 | 19.29 |
| | | Vanilla w/ Role | 30.35 | 18.70 | 12.46 | 17.06 | 13.88 | 14.02 | 17.75 |
| | Qwen2-72B-Instruct | Vanilla w/o Role | 33.64 | 20.95 | 13.05 | 26.13 | 16.72 | 16.04 | 21.09 |
| | | Vanilla w/ Role | 33.24 | 20.45 | 12.56 | 26.08 | 15.64 | 16.48 | 20.74 |
| | GPT-4o-mini | Vanilla w/o Role | 35.29 | 22.26 | 14.15 | 27.66 | 17.29 | 17.04 | 22.28 |
| | | Vanilla w/ Role | 35.48 | 22.43 | 14.25 | 28.81 | 18.32 | 17.53 | 22.80 |
| | GPT-4o | Vanilla w/o Role | 34.27 | 21.56 | 11.96 | 22.93 | 13.54 | 14.41 | 19.78 |
| | | Vanilla w/ Role | 34.22 | 20.88 | 11.96 | 22.43 | 13.59 | 14.33 | 19.57 |
| | GPT-4o* | Vanilla w/o Role | 33.70 | 21.17 | 14.05 | 25.00 | 18.37 | 16.84 | 21.52 |
| | | Vanilla w/ Role | 32.41 | 21.01 | 15.65 | 27.19 | 22.07 | 20.40 | 23.12 |
| 10 | T5-base | PromptRank | 37.88 | **37.76** | 20.66 | **31.59** | 20.13 | 16.71 | 27.46 |
| | Llama3-8B-Instruct | Vanilla w/o Role | 46.42 | 34.95 | 22.46 | 27.74 | 25.30 | 20.58 | 29.57 |
| | | Vanilla w/ Role | 46.24 | 35.75 | **22.92** | 28.65 | 25.04 | 20.86 | **29.91** |
| | Llama3-70B-Instruct | Vanilla w/o Role | 45.82 | 34.03 | 20.17 | 28.82 | 21.76 | 20.02 | 28.44 |
| | | Vanilla w/ Role | **47.05** | 35.08 | 21.33 | 31.54 | 22.02 | 20.66 | 29.61 |
| | Gemma2-9b-it | Vanilla w/o Role | 42.66 | 31.34 | 20.53 | 26.17 | 25.21 | 22.52 | 28.07 |
| | | Vanilla w/ Role | 42.16 | 31.78 | 20.85 | 26.94 | **25.32** | **22.76** | 28.30 |
| | Gemma2-27b-it | Vanilla w/o Role | 45.16 | 33.21 | 20.63 | 29.58 | 24.35 | 21.79 | 29.12 |
| | | Vanilla w/ Role | 44.61 | 33.68 | 21.08 | 30.34 | 25.16 | 21.71 | 29.43 |
| | Qwen2-7B-Instruct | Vanilla w/o Role | 39.67 | 30.39 | 17.03 | 24.29 | 16.32 | 15.89 | 23.93 |
| | | Vanilla w/ Role | 38.75 | 28.62 | 16.40 | 18.00 | 16.10 | 15.09 | 22.16 |
| | Qwen2-72B-Instruct | Vanilla w/o Role | 43.10 | 32.32 | 17.56 | 27.17 | 18.55 | 17.45 | 26.03 |
| | | Vanilla w/ Role | 42.78 | 31.49 | 17.29 | 27.05 | 17.81 | 17.17 | 25.60 |
| | GPT-4o-mini | Vanilla w/o Role | 45.49 | 33.30 | 19.07 | 30.72 | 19.56 | 17.95 | 27.68 |
| | | Vanilla w/ Role | 45.02 | 33.62 | 18.68 | 31.19 | 20.78 | 18.75 | 28.01 |
| | GPT-4o | Vanilla w/o Role | 43.40 | 33.16 | 16.45 | 26.19 | 16.45 | 15.94 | 25.27 |
| | | Vanilla w/ Role | 43.08 | 32.01 | 16.52 | 25.36 | 16.32 | 15.63 | 24.82 |
| | GPT-4o* | Vanilla w/o Role | 44.41 | 33.05 | 19.11 | 29.47 | 21.35 | 18.06 | 27.58 |
| | | Vanilla w/ Role | 43.09 | 32.21 | 20.75 | 29.93 | 24.94 | 21.79 | 28.79 |
| 15 | T5-base | PromptRank | 38.17 | **41.57** | 21.35 | **31.01** | 20.12 | 16.02 | 28.04 |
| | Llama3-8B-Instruct | Vanilla w/o Role | 48.16 | 38.56 | 22.89 | 25.09 | 23.35 | 18.32 | 29.39 |
| | | Vanilla w/ Role | 48.04 | 39.68 | **23.21** | 25.79 | 23.53 | 18.27 | **29.75** |
| | Llama3-70B-Instruct | Vanilla w/o Role | 46.89 | 36.64 | 20.80 | 27.29 | 20.91 | 18.17 | 28.45 |
| | | Vanilla w/ Role | **48.56** | 38.34 | 21.96 | 29.45 | 21.03 | 18.30 | 29.61 |
| | Gemma2-9b-it | Vanilla w/o Role | 43.20 | 33.53 | 20.68 | 24.98 | **24.66** | **21.36** | 28.07 |
| | | Vanilla w/ Role | 42.63 | 33.66 | 21.09 | 25.69 | 24.40 | 21.33 | 28.13 |
| | Gemma2-27b-it | Vanilla w/o Role | 45.82 | 36.04 | 21.12 | 27.65 | 23.87 | 20.23 | 29.12 |
| | | Vanilla w/ Role | 45.20 | 36.43 | 20.94 | 28.60 | 24.41 | 20.16 | 29.29 |
| | Qwen2-7B-Instruct | Vanilla w/o Role | 42.07 | 35.95 | 17.73 | 22.51 | 16.74 | 14.94 | 24.99 |
| | | Vanilla w/ Role | 40.90 | 34.13 | 17.93 | 17.55 | 16.23 | 14.56 | 23.55 |
| | Qwen2-72B-Instruct | Vanilla w/o Role | 45.68 | 38.19 | 18.68 | 25.57 | 18.24 | 16.00 | 27.06 |
| | | Vanilla w/ Role | 45.61 | 37.46 | 18.70 | 25.83 | 17.75 | 15.94 | 26.88 |
| | GPT-4o-mini | Vanilla w/o Role | 47.64 | 39.41 | 20.41 | 28.48 | 18.53 | 16.16 | 28.44 |
| | | Vanilla w/ Role | 47.61 | 39.61 | 20.12 | 28.85 | 19.56 | 17.08 | 28.81 |
| | GPT-4o | Vanilla w/o Role | 46.20 | 39.60 | 18.68 | 25.32 | 16.89 | 15.66 | 27.06 |
| | | Vanilla w/ Role | 45.87 | 38.49 | 18.41 | 24.63 | 17.19 | 15.43 | 26.67 |
| | GPT-4o* | Vanilla w/o Role | 48.20 | 39.32 | 21.64 | 28.15 | 21.74 | 17.72 | 29.46 |
| | | Vanilla w/ Role | 45.49 | 38.06 | 21.98 | 27.58 | 23.24 | 19.26 | 29.27 |

Table 10: Performance comparison (F1@K) of various LLMs on keyphrase extraction across datasets using vanilla prompting with and without role prompting. GPT-4o* uses 'keyword' instead of 'keyphrase' in prompts. **Bold**: best performance per dataset and K.

| F1@K | Model | Method | Dataset | | | | | | AVG |
|---|---|---|---|---|---|---|---|---|---|
| | | | Inspec | SemEval2017 | SemEval2010 | DUC2001 | NUS | Krapivin | |
| 5 | T5-base | PromptRank | 31.73 | **27.14** | 17.24 | 27.39 | 17.24 | 16.11 | 22.81 |
| | Llama3-8B-Instruct | Vanilla | 36.75 | 24.21 | 17.04 | 27.60 | 22.70 | 21.45 | 24.96 |
| | | Hybrid | 36.81 | 25.12 | 18.44 | 29.98 | 23.27 | 21.41 | 25.84 |
| | Llama3-70B-Instruct | Vanilla | 36.93 | 23.67 | 16.64 | 28.96 | 19.68 | 19.91 | 24.30 |
| | | Hybrid | **37.35** | 25.19 | **18.54** | **33.02** | 21.67 | 21.53 | **26.22** |
| | Gemma2-9b-it | Vanilla | 35.14 | 22.10 | 16.84 | 25.95 | 23.27 | 22.43 | 24.29 |
| | | Hybrid | 35.28 | 22.86 | 17.84 | 28.09 | **24.06** | **23.53** | 25.28 |
| | Gemma2-27b-it | Vanilla | 36.30 | 23.21 | 16.94 | 28.94 | 22.58 | 21.75 | 24.95 |
| | | Hybrid | 36.72 | 23.75 | 17.54 | 30.28 | 23.15 | 21.42 | 25.48 |
| | Qwen2-7B-Instruct | Vanilla | 31.41 | 19.61 | 12.46 | 23.08 | 14.45 | 14.75 | 19.29 |
| | | Hybrid | 30.33 | 18.92 | 12.56 | 18.21 | 13.99 | 14.22 | 18.04 |
| | Qwen2-72B-Instruct | Vanilla | 33.64 | 20.95 | 13.05 | 26.13 | 16.72 | 16.04 | 21.09 |
| | | Hybrid | 33.92 | 21.63 | 13.55 | 27.88 | 16.84 | 17.04 | 21.81 |
| | GPT-4o-mini | Vanilla | 35.29 | 22.26 | 14.15 | 27.66 | 17.29 | 17.04 | 22.28 |
| | | Hybrid | 35.48 | 22.43 | 14.35 | 28.76 | 18.32 | 17.65 | 22.83 |
| | GPT-4o | Vanilla | 34.27 | 21.56 | 11.96 | 22.93 | 13.54 | 14.41 | 19.78 |
| | | Hybrid | 34.21 | 20.90 | 11.86 | 22.83 | 13.71 | 14.41 | 19.65 |
| | GPT-4o* | Vanilla | 33.70 | 21.17 | 14.05 | 25.00 | 18.37 | 16.84 | 21.52 |
| | | Hybrid | 33.81 | 21.99 | 13.65 | 27.99 | 18.26 | 17.12 | 22.14 |
| 10 | T5-base | PromptRank | 37.88 | **37.76** | 20.66 | 31.59 | 20.13 | 16.71 | 27.46 |
| | Llama3-8B-Instruct | Vanilla | 46.42 | 34.95 | **22.46** | 27.74 | 25.30 | 20.58 | 29.57 |
| | | Hybrid | 45.80 | 36.08 | 22.45 | 29.55 | 24.99 | 21.09 | 29.99 |
| | Llama3-70B-Instruct | Vanilla | 45.82 | 34.03 | 20.17 | 28.82 | 21.76 | 20.02 | 28.44 |
| | | Hybrid | **46.94** | 36.23 | 22.19 | **34.19** | 23.45 | 20.22 | **30.54** |
| | Gemma2-9b-it | Vanilla | 42.66 | 31.34 | 20.53 | 26.17 | 25.21 | 22.52 | 28.07 |
| | | Hybrid | 42.90 | 33.17 | 21.79 | 28.77 | **25.55** | **22.73** | 29.15 |
| | Gemma2-27b-it | Vanilla | 45.16 | 33.21 | 20.63 | 29.58 | 24.35 | 21.79 | 29.12 |
| | | Hybrid | 45.59 | 35.33 | 21.93 | 31.05 | 25.26 | 21.90 | 30.18 |
| | Qwen2-7B-Instruct | Vanilla | 39.67 | 30.39 | 17.03 | 24.29 | 16.32 | 15.89 | 23.93 |
| | | Hybrid | 38.35 | 28.78 | 16.35 | 19.57 | 15.97 | 15.03 | 22.34 |
| | Qwen2-72B-Instruct | Vanilla | 43.10 | 32.32 | 17.56 | 27.17 | 18.55 | 17.45 | 26.03 |
| | | Hybrid | 44.27 | 33.00 | 17.65 | 28.94 | 18.51 | 17.57 | 26.66 |
| | GPT-4o-mini | Vanilla | 45.49 | 33.30 | 19.07 | 30.72 | 19.56 | 17.95 | 27.68 |
| | | Hybrid | 45.00 | 33.63 | 18.83 | 31.29 | 20.87 | 18.89 | 28.09 |
| | GPT-4o | Vanilla | 43.40 | 33.16 | 16.45 | 26.19 | 16.45 | 15.94 | 25.27 |
| | | Hybrid | 43.08 | 32.19 | 16.67 | 25.87 | 16.50 | 15.72 | 25.01 |
| | GPT-4o* | Vanilla | 44.41 | 33.05 | 19.11 | 29.47 | 21.35 | 18.06 | 27.58 |
| | | Hybrid | 44.61 | 34.23 | 19.15 | 32.40 | 21.44 | 18.75 | 28.43 |
| 15 | T5-base | PromptRank | 38.17 | **41.57** | 21.35 | 31.01 | 20.12 | 16.02 | 28.04 |
| | Llama3-8B-Instruct | Vanilla | 48.16 | 38.56 | 22.89 | 25.09 | 23.35 | 18.32 | 29.39 |
| | | Hybrid | 45.17 | 40.23 | 22.87 | 27.21 | 23.70 | 18.35 | 29.59 |
| | Llama3-70B-Instruct | Vanilla | 46.89 | 36.64 | 20.80 | 27.29 | 20.91 | 18.17 | 28.45 |
| | | Hybrid | 47.83 | 41.16 | **23.71** | 31.61 | 22.29 | 18.48 | **30.85** |
| | Gemma2-9b-it | Vanilla | 43.20 | 33.53 | 20.68 | 24.98 | 24.66 | **21.36** | 28.07 |
| | | Hybrid | 43.92 | 36.50 | 22.57 | 28.96 | 24.88 | 21.00 | 29.64 |
| | Gemma2-27b-it | Vanilla | 45.82 | 36.04 | 21.12 | 27.65 | 23.87 | 20.23 | 29.12 |
| | | Hybrid | 46.68 | 39.50 | 22.96 | 30.46 | **25.09** | 20.36 | 30.84 |
| | Qwen2-7B-Instruct | Vanilla | 42.07 | 35.95 | 17.73 | 22.51 | 16.74 | 14.94 | 24.99 |
| | | Hybrid | 39.54 | 34.25 | 17.73 | 18.93 | 15.65 | 14.55 | 23.44 |
| | Qwen2-72B-Instruct | Vanilla | 45.68 | 38.19 | 18.68 | 25.57 | 18.24 | 16.00 | 27.06 |
| | | Hybrid | 45.97 | 39.44 | 19.77 | 27.18 | 18.85 | 16.53 | 27.96 |
| | GPT-4o-mini | Vanilla | 47.64 | 39.41 | 20.41 | 28.48 | 18.53 | 16.16 | 28.44 |
| | | Hybrid | 46.38 | 39.64 | 20.49 | 29.11 | 19.62 | 17.08 | 28.72 |
| | GPT-4o | Vanilla | 46.20 | 39.60 | 18.68 | 25.32 | 16.89 | 15.66 | 27.06 |
| | | Hybrid | 44.70 | 38.87 | 18.82 | 25.69 | 17.53 | 15.74 | 26.89 |
| | GPT-4o* | Vanilla | **48.20** | 39.32 | 21.64 | 28.15 | 21.74 | 17.72 | 29.46 |
| | | Hybrid | 48.06 | 41.36 | 22.33 | **31.89** | 22.08 | 18.59 | 30.72 |

Table 11: Performance comparison (F1@K) of various LLMs on keyphrase extraction across datasets using vanilla and hybrid prompting. GPT-4o*: 'keyword' used instead of 'keyphrase'. **Bold**: best score per dataset and K.