Enhancing Frame Detection with Retrieval Augmented Generation

Papa Abdou Karim Karou Diallo$^{\diamond\dagger\star}$ Amal Zouaq$^{\diamond\dagger\star}$

$^{\diamond}$LAMA-WeST †Polytechnique Montreal *Mila

Abstract. Recent advancements in Natural Language Processing have significantly improved the extraction of structured semantic representations from unstructured text, especially through Frame Semantic Role Labeling (FSRL). Despite this progress, the potential of Retrieval-Augmented Generation (RAG) models for frame detection remains under-explored. In this paper, we present the first RAG-based approach for frame detection called **RCIF** (**R**etrieve **C**andidates and **I**dentify **F**rames). RCIF is also the first approach to operate without the need for explicit target span and comprises three main stages: (1) generation of frame embeddings from various representations ; (2) retrieval of candidate frames given an input text; and (3) identification of the most suitable frames. We conducted extensive experiments across multiple configurations, including zero-shot, few-shot, and fine-tuning settings. Our results show that our retrieval component reduces the complexity of the task by narrowing the search space thus allowing the frame identifier to refine and complete the set of candidates. Our approach achieves state-of-the-art performance on FrameNet 1.5 and 1.7, demonstrating its robustness in scenarios where only raw text is provided.

Keywords: Frame semantic parsing, RAG, LLMs.

1 Introduction

Large Language Models (LLMs) have led to major advancements in natural language understanding, achieving state-of-the-art performance across a range of tasks. However, despite their impressive capabilities, LLMs often show sensitivity to input phrasing and struggle to generalize across different lexical variations of semantically equivalent inputs [12,14,22]. This brittleness limits their robustness and usability, especially in downstream applications that require precise and structured interpretations of language, such as semantic parsing or knowledge base querying. To address these limitations, we hypothesize that grounding natural language understanding in structured semantic representations is essential. We investigate the research question: *how can we make models have the same representation for various question reformulations ?*. One method is to bring all the formulations to a single representation. In this work, we investigate frame semantics as such representation.

Frames represent prototypical situations or events (e.g., *Buying*, *Travel*, *Communication*). A FrameNet frame f is defined by (1) its *label* representing its name, (2) its *description* that provides a comprehensive explanation of its semantics, (3) its set of *frame-elements (FEs)*, which serve as its core semantic components, capturing contextual and relational information about the frame and finally (4) its set of *lexical units (LUs)*, consisting of lemmas paired with their parts of speech, which denote the frame or specific aspects of it. Within a sentence, tokens (words or phrases) that evoke a frame are referred to as *targets*. To get a frame-based structured representation, the first step is to detect the frames evoked by the question (sentence or text in general). It involves selecting the correct frame for a given target word or phrase (called the *target span*) in a text. For example, in the question *"Where did she buy the book?"*, the verb *buy* evokes the *frame* named *Commerce_buy*, with associated roles like *Buyer*, *Goods*, *Means*, *Money*, *Rate* and *Seller* that are among the *frame-elements* of the frame *Commerce_buy*. Traditional frame detection

David C. Wyld et al. (Eds): COMIT, AISO, CRBL, WIMNET, EDUPT, NLDM – 2025
pp. 123-134, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151210

approaches assume that the target word is explicitly specified in advance [5,16,7,18,1]. However, in many real-world scenarios, there is no predefined *target span*. This lack of explicit alignment makes these target-based frame detection approaches less practical for knowledge base querying tasks. This limitation motivates the need for a target-free frame detection, where the model must infer relevant frames directly from the full input text without being told which words evoke them. Achieving this requires both efficient candidate retrieval and accurate frame selection mechanisms, especially when dealing with large frame inventories and varied lexical expressions.

In this paper, we propose a new method, called **RCIF** (**R**etrieve **C**andidates and **I**dentify **F**rames), that addresses these challenges by combining structured semantic representations with a Retrieval-Augmented Generation (RAG) architecture. As depicted in Figure 1, RCIF operates in three stages: (1) a semantic index is constructed from embeddings of frames; (2) given an input question, the model retrieves the most relevant candidate frames from this index; (3) a generative LLM selects the best-matching frames from the retrieved candidates, effectively acting as a classifier over retrieved options.

The main contributions of this paper are:

- We present the first method that leverages a RAG-based framework for frame detection using a generative LLM.
- We introduce a novel approach for target-free frame detection, making frame-semantic parsing more broadly applicable in real-world scenarios where explicit targets are not available.

Overall, our approach not only enhances frame detection accuracy but also addresses the challenge of target-free input by dynamically narrowing the search space to relevant frames, improving both recall and precision in frame identification.

2 Related Works

The literature on frame parsing can be broadly divided into two main approaches: methods that frame the task as a sequence-to-sequence (Seq2Seq) generation problem and methods based on representation learning.

Seq2Seq approaches [17,15,9,2] define frame parsing as a generative task, decomposing it into subtasks such as trigger identification, frame classification, and argument extraction. These methods leverage pre-trained language models and task-specific optimizations to balance the subtask distributions and mitigate data scarcity [9,2]. As illustration, models like T5 [15] are pre-trained on PropBank [11,10] and FrameNet exemplars, with text augmentation techniques applied to improve robustness and FrameNet lexical units incorporated to enhance frame classification accuracy. Then, they are fine-tuned on each of the aforementioned sub-tasks. For exemple, [9] adopt a shared encoder with specialized decoders for each sub-task, enabling the model to leverage a common representation while handling each task independently within the same architecture. The Seq2Seq methods share a focus on utilizing the flexibility of generative models to capture the sequential nature of frame parsing tasks.

Representation learning approaches, in contrast, focus on constructing enriched embeddings that align sentence-level context (or just the target span) with candidate frames [5,7]. They often employ graph-based techniques, such as Graph Neural Networks (GNNs) [19], or contrastive learning [8] to incorporate external knowledge and enhance the robustness of frame representations. These methods also emphasize semantic alignment through embedding techniques that integrate knowledge from FrameNet's structure. Graph-based methods,

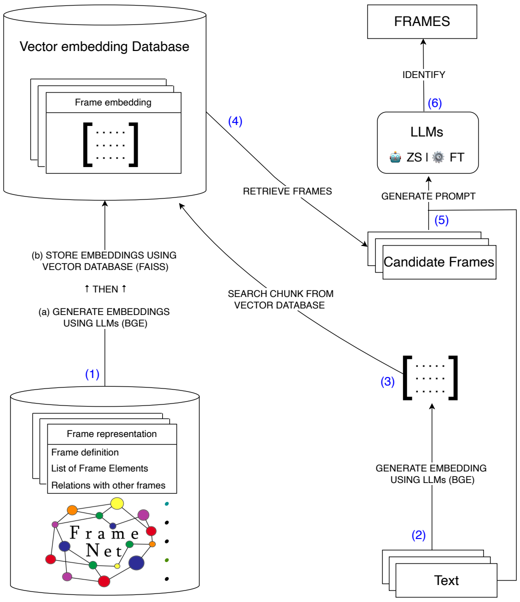

Vector embedding Database
Frame embedding
FRAMES
IDENTIFY
(6)
LLMs
ZS I FT
(4)
RETRIEVE FRAMES
GENERATE PROMPT
(5)
Candidate Frames
(b) STORE EMBEDDINGS USING VECTOR DATABASE (FAISS)
↑ THEN ↑
(a) GENERATE EMBEDDINGS USING LLMs (BGE)
SEARCH CHUNK FROM VECTOR DATABASE
(1)
(3)
Frame representation
Frame definition
List of Frame Elements
Relations with other frames
GENERATE EMBEDDING USING LLMs (BGE)
Frame Net
(2)
Text

Fig. 1. Overview of our proposed method called **RCIF** (**R**etrieve **C**andidates and **I**dentify **F**rames). (1) Frame embeddings are generated using an embedding model based on various frame representations. These embeddings are stored in a vector database. (2-3) Given an input text, the system retrieves candidate frames based on similarity scores of input text and frames embeddings. (4-5) An LLM is then fine-tuned with dynamic prompts to select the best matching frames from the retrieved candidates, completing the identification process.

for instance, exploit relationships between frames and frame elements [16,21,18], while contrastive learning approaches align contextual representations of target span with frame embeddings to refine predictions [5,7,1]. More specifically, the previous SOTA approaches we compare our results with (KGFI[16] and CoFFTEA[1]) use representation learning strategies for frame identification. KGFI incorporates structured frame knowledge—such as definitions, frame elements, and inter-frame relations—into a joint embedding space to better align targets and frames via dot-product similarity. CoFFTEA, on the other hand, employs contrastive learning with dual encoders and a coarse-to-fine curriculum to model target-frame alignment. A key limitation that hinders these approaches from generalizing effectively and being applicable in real-world scenarios is their reliance on both the context (text/sentence) and a target, which has to be specified at input. For instance, consider the sentence: *"We help people train for and find jobs that make it possible for them to get off of welfare."*. To detect the frame *"Assistance,"* these approaches require information about the position of the target span *"help"*. This dependency restricts their flexibility and reduces their practical utility in settings where only raw text without predefined targets is available.

To address the computational challenges in frame detection, we propose an approach to reduce the search space by first retrieving a subset of potential candidate frames that are likely to be evoked by the sentence. By limiting the frame search space, we aim to decrease the number of frame evaluations for each word, thereby reducing overall complexity, but we also strive to maintain a high recall among the most likely potential frames.

3 Methodology

3.1 Datasets

To ensure a fair comparison with prior work, we assess our model's performance on FrameNet 1.5, adhering to the original train/dev/test data splits established by [3]. Additionally, we extend our evaluation to FrameNet 1.7, released in 2016, which offers approximately 20% more gold-standard annotations compared to FrameNet 1.5. For both FrameNet 1.5 and FrameNet 1.7, we follow the data splits defined by An et al. [1]. Table 1 provides details on the number of examples in each split, including the exemplars dataset commonly used for pretraining.

Table 1. Dataset Distribution for FrameNet 1.5 and FrameNet 1.7

Dataset Split	**FrameNet 1.5**		**FrameNet 1.7**	
	All	**Uniques**	**All**	**Uniques**
Train	16,621	2,653	19,391	3,353
Dev	2,284	326	2,272	326
Test	4,428	875	6,714	1,247
Exemplars	153,946	147,483	192,431	168,266
# Frames	1,019		1,221	

3.2 Concepts definition and task description

As described in the introduction, a FrameNet frame f is defined by its label, its textual description, its set of frame-elements (FEs), and a set of lexical units (LUs).

We formulate our task as the detection of frames $f_i, \ldots, f_j$ within a sentence $S = w_1, \ldots, w_n$. Unlike approaches such as CoffTea [1] and related works that rely on both a sentence and a specific target span $t = w_{t_s}, \ldots, w_{t_e}$ (with w_{t_s} and w_{t_e} respectively corresponding to the start and the end of the target span) to identify a single frame, our model, **RCIF**, is designed to identify frames in the absence of such target span information.

FrameNet datasets consist of entries such as the sentence *"I was sad when I could n't go to the snack bar to buy a soda."* where the underlined span is the target whose position is provided as well as the frame which is "Commerce_buy" and its definition. Thus, the entry is repeated as many times as there are target spans or possible frames for the same sentence with, for every instance, a new couple of target span and frame.

To generalize over target-less raw text, we adapt the original datasets by grouping frames by sentence so that each unique sentence is associated with all the frames it evokes. For instance, the six occurrences of the previous sentence are merged into a single entry, discarding information about the target and retaining only the following list of possible frames such as "Emotion_directed", "Capability", "Likelihood", "Locative_relation", "Goal", "Commerce_buy" and "Temporal_collocation". As one can see, this new target-free dataset formulation constitutes a harder task than the previous one. We thus address this problem by first retrieving a set of candidates and then identifying the appropriate frames, as detailed in Section 3.3.

As shown in Table 2, the resulting data distribution includes a minimum of 1 frame per sentence, a maximum of 24 frames per sentence, and an average of 5 frames per sentence for FrameNet 1.5. For FrameNet 1.7, we observe similar statistics, with a minimum of 1 frame, a maximum of 23 frames, and an average of 5 frames per sentence. The exemplar data presents an average and minimum of one frame per sentence across both datasets, with a maximum of 14 frames for FrameNet 1.5 and 21 frames for FrameNet 1.7.

Table 2. Number of frames per sentence in the different splits for FrameNet 1.5 and FrameNet 1.7.

Dataset Split	FrameNet 1.5			FrameNet 1.7		
	Min	Max	Mean	Min	Max	Mean
Train	1	21	5	1	21	5
Validation	1	18	6	1	18	6
Test	1	24	4	1	23	4
Exemplars	1	14	1	1	21	1

Our general architecture consists of three steps: (1) generation of frame embeddings from various representations (section 3.3); (2) retrieval of candidate frames given an input text (section 3.3); and (3) identification of the best frames (section 3.4) using the Llama 3.2-3B model [4].

3.3 Frames representations and retrieval of candidates frames

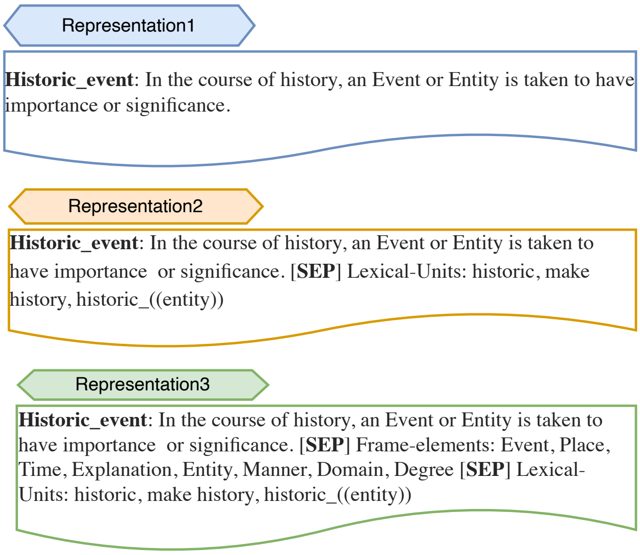
In this phase, we employ a frozen-RAG model to facilitate candidate retrieval. For the retrieval component, we compare different frame representations using labels, descriptions, lexical units (LUs), or frame elements (FEs). Figure 2 shows different representations of the frame "Historic_event". We systematically explore all these options, generating embeddings for each frame representation, which are then stored in a vector database. When processing a new text, we generate its vector embedding. For embedding generation, we utilize the English version of BGE[1] [13,20]. The model employs a BERT-like architecture, utilizing the hidden state of the [CLS] token as the embedding. We then perform a similarity search using FAISS to retrieve the top k candidates based on similarity scores between the question and the frame embeddings.

3.4 Identification of frames

Identifying the set of frames evoked by a sentence without a specified target span presents significant complexity influenced by sentence length and the total number of frames in the lexicon (1019 in FrameNet 1.5 and 1221 in FrameNet 1.7). Our previous step (in section 3.3)reduces the search space by selecting a list of potential frame candidates that guide the model towards the correct frames. We then use a pretrained LLM to finalize frame selection. The LLM is fine-tuned in an *Instruction-Input-Output* format. This prompt design allows the fine-tuned model to not only gain inspiration from the retrieved candidates but also to consider frames identified in previous batches, thereby accommodating retrieval imperfections where some correct frames may not appear in the candidates list. The prompt is provided in Table 3.

We conducted multiple experiments using several models such as Llama 3.2-3B model [4], Llama 3.1-8B, Phi-4, and Qwen2.5-7B-Instruct across three main settings: zero-shot, few-shot, and fine-tuning but we just keep the best performing model (Llama 3.2-3B). For each setting, we implemented two configurations: one that explicitly indicates the number

[1] https://huggingface.co/BAAI



Historic_event: In the course of history, an Event or Entity is taken to have importance or significance.

Historic_event: In the course of history, an Event or Entity is taken to have importance or significance. [**SEP**] Lexical-Units: historic, make history, historic_((entity))

Historic_event: In the course of history, an Event or Entity is taken to have importance or significance. [**SEP**] Frame-elements: Event, Place, Time, Explanation, Entity, Manner, Domain, Degree [**SEP**] Lexical-Units: historic, make history, historic_((entity))

Fig. 2. Different representations of frames used in the retrieval component. *Representation1* consists of the frame label and its textual description. *Representation2* extends the previous one by appending a list of lexical units, while *Representation3* further enriches *Representation2* by incorporating a list of frame elements, resulting in a more comprehensive one.

of frames the model is expected to detect, and another that does not. This distinction allows us to test the hypothesis that, without specifying the number of frames, the model may struggle to accurately determine the appropriate number of candidates to select. Additionally, we included a baseline experiment that involves fine-tuning the model to generate frames without leveraging the retrieval component as an initial step. Technical details about the model fine-tuning using LoRA [6] 4-bit Quantization and SFTTrainer[2] are provided in Table 4.

4 Results

4.1 Retrieval component

Table 5 presents the performance of candidate frames retrieval using the English version of the BGE embedding model [13,20]. Retrieval is configured to select $K = \max_{\text{frames}} = 24$ candidates, which corresponds to the maximum number of frames a sentence might evoke. This setting is chosen to maximize recall, ensuring that the subsequent detection/identification stage has a high likelihood of finding relevant frames among the candidate set.

As shown in Table 5, the third frame representation (*Representation 3*, including frame label and description, the list of frame-elements and lexical-units), yields on average the highest recall, highlighting it as the most effective representation format, although very close to *Representation 2*. Across all datasets and frame representations, precision remains low, a consequence of maximizing recall by retrieving a surplus of frames ($\max_{\text{frames}} = 24$) compared to the average need ($\text{avg}_{\text{frames}} = 5$). This retrieval strategy not only provides the frame detector component with the broadest possible set of relevant candidates, but it also encourages the model, during fine-tuning, to rely less on parametric memory and more on generalization, enhancing its robustness.

[2] https://huggingface.co/docs/trl/en/sft_trainer

Table 3. Example of the dynamic prompt used to fine-tune the LLM

Instruction	You are an assistant tasked with identifying the relevant frames evoked by a given sentence. You will get a list of potential candidate frames, but not all possible frames will always be included. If a relevant frame is not in the provided list, use your knowledge and prior context to identify the most relevant one(s). Give an answer like: `Response: frames: [frame_name, . . . , frame_name]`, with no additional discursive or explanatory text.
Input	Sentence: Bush said that it was Khan who sold centrifuges to North Korea. frame name: `Commerce_sell` – frame definition: These are words describing basic commercial transactions involving a buyer and a seller exchanging money and goods, taking the perspective of the seller. frame name: `Statement` – frame definition: This frame contains verbs and nouns that communicate the act of a Speaker to address a Message to some Addressee using language.
Output	`['Commerce_sell', 'Statement']`

Table 4. Technical details of the fine-tuning

Parameters	**Values**
GPU model	A100-40gb
Number of hours (training and inference)	12
Number of epochs	10
Max Sequence Length	2048
Packing	False (for faster training)
Per Device Batch Size	16
Gradient Accumulation Steps	4
Warmup Steps	5
Learning Rate	2e-4
Precision Mode	fp16 or bf16 (conditional)
Logging Steps	1
Optimizer	adamw_8bit
Weight Decay	0.01
Learning Rate Scheduler	Linear
Random Seed	3407
Evaluation Strategy	Epoch

4.2 Frame detection

Table 6 presents the frame detection results of the Llama 3.2 - 3B model fine-tuned for 10 epochs on the complete training sets of FrameNet 1.5 and FrameNet 1.7. While the best accuracy is reported in [18] and COFFTEA [1] for FrameNet 1.5, our model achieves higher precision/recall, outperforming prior work by approximately 4 points in recall. This improvement is attributed to the reduction in search space during the retrieval phase and effective de-noising (elimination of irrelevant candidates) by the fine-tuned LLM. Starting with a retrieval phase precision of 5% and a recall of 89% as shown in Table 5, our final model enhances both metrics to around 92%, indicating successful removal of incorrect candidates while retaining relevant ones. This effect is even more pronounced in FrameNet 1.7, where our model achieves top performance across all metrics, with a precision of 99% and a recall of 97%, demonstrating that the model effectively filters out incorrect candidates. The additional training samples in FrameNet 1.7 (26% more than FrameNet 1.5) further contribute to this improvement. Interestingly, not specifying the exact number of frames to generate didn't hurt performance—possibly because of how the data is distributed. This finding is crucial for real-world applications, as providing exact frame counts is often infeasible with out-of-distribution data. Consistent with the task description in section 3.2,

Table 5. Retrieval Performances on FrameNet 1.5 and FrameNet 1.7 train sets (%)

	FrameNet 1.5		**FrameNet 1.7**		**Average**	
Metrics	**P**	**R**	**P**	**R**	**P**	**R**
Rep1	4	79	3	72	4	76
Rep2	5	90	4	77	5	84
Rep3	5	89	4	81	5	**85**

R : Recall
P : Precision
Rep_i : Representation_i with $i \in \{1, 2, 3\}$

this framework is designed for practical deployment where only the sentence is provided without target or frame count specifications. Results of experiments where exemplars data from FrameNet 1.5 and 1.7 are used as additional training data are provided in Appendix A.1.

Table 6. Performance (%) on FrameNet 1.5 and FrameNet 1.7

Approach	**Accuracy**	**Precision**	**Recall**
FrameNet 1.5			
KGFI (2021) [16]	92	-	86
Tamburini [18]	**93**	-	-
COFFTEA [1]	**93**	-	88
Baseline (Fine-Tuning without retrieval of candidates)	24	34	43
RCIF (Zero-Shot) [without/with information about the number of Gold Frames]	12 / 13	12 / 18	50 / 33
RCIF (Few-Shot) [without/with information about the number of Gold Frames]	16 / 17	24 / 26	42 / 33
RCIF (Fine-Tuning) [without/with information about the number of Gold Frames]	89 / 92	91 / **92**	**92** / **92**
FrameNet 1.7			
KGFI (2021) [16]	92	-	86
Tamburini [18]	92	-	-
COFFTEA [1]	93	-	87
Baseline (Fine-Tuning without retrieval of candidates)	25	34	44
RCIF (Zero-Shot) [without/with information about the number of Gold Frames]	12 / 13	12 / 18	50 / 33
RCIF (Few-Shot) [without/with information about the number of Gold Frames]	16 / 17	24 / 26	42 / 33
RCIF (Fine-Tuning) [without/with information about the number of Gold Frames]	**95** / 94	**99** / 96	**97** / **97**

5 Conclusion

In this paper, we introduced a novel and simple approach called **RCIF** (**R**etrieve **C**andidates and **I**dentify **F**rames) to frame detection leveraging RAG models. Unlike previous SOTA methods, which rely on predefined spans within the input text for frame detection, our method operates solely on the input text sequence without requiring additional information about the target span. Our proposed pipeline consists of two components: a candidate retriever and an LLM that selects the correct frames from the retrieved set of candidates. This approach demonstrated improved performance over SOTA methods on FrameNet 1.5 and achieved higher recall (10 additional points) on FrameNet 1.7, which provides a larger training set. Consequently, our method is well-suited to real-world applications where only raw text is available, and specific spans for frame detection are not predefined.

Limitations

This study is limited to the use of "frozen-RAG" for the retrieval component. Exploring trained versions of RAG could be a promising direction for future research. Additionally, our current representation associates a text to a bag of frames and frame-elements. Our future efforts will focus on extracting full semantic representations with a mapping of arguments to their corresponding frame-elements. We also plan to experiment if such an approach could be leveraged in knowledge base querying by providing a similar representation for various question formulations.

Acknowledgments

We are grateful to the NSERC Discovery Grant Program and the IVADO R^{3}AI NLP Regroupement (theme 3), which have funded this research. The authors would also like to express their gratitude to Compute Canada (Calcul Quebec) for providing computational resources.

References

1. Kaikai An, Ce Zheng, Bofei Gao, Haozhe Zhao, and Baobao Chang. Coarse-to-fine dual encoders are better frame identification learners. *arXiv preprint arXiv:2310.13316*, 2023.
2. David Chanin. Open-source frame semantic parsing. *arXiv preprint arXiv:2303.12788*, 2023.
3. Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. Frame-semantic parsing. *Computational linguistics*, 40(1):9–56, 2014.
4. Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
5. Silvana Hartmann, Ilia Kuznetsov, M Teresa Martín-Valdivia, and Iryna Gurevych. Out-of-domain framenet semantic role labeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 471–482, 2017.
6. Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
7. Tianyu Jiang and Ellen Riloff. Exploiting definitions for frame identification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2429–2434, 2021.
8. Wei Ju, Yifan Wang, Yifang Qin, Zhengyang Mao, Zhiping Xiao, Junyu Luo, Junwei Yang, Yiyang Gu, Dongjie Wang, Qingqing Long, et al. Towards graph contrastive learning: A survey and beyond. *arXiv preprint arXiv:2405.11868*, 2024.
9. Aditya Kalyanpur, Or Biran, Tom Breloff, Jennifer Chu-Carroll, Ariel Diertani, Owen Rambow, and Mark Sammons. Open-domain frame semantic parsing using transformers. *arXiv preprint arXiv:2010.10998*, 2020.
10. Paul Kingsbury and Martha Palmer. Propbank: the next level of treebank. In *Proceedings of Treebanks and lexical Theories*, volume 3. Citeseer, 2003.
11. Paul R Kingsbury and Martha Palmer. From treebank to propbank. In *LREC*, pages 1989–1993, 2002.
12. Alina Leidinger, Robert Van Rooij, and Ekaterina Shutova. The language of prompting: What linguistic properties make a prompt successful? *arXiv preprint arXiv:2311.01967*, 2023.
13. Chaofan Li, MingHao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Yingxia Shao, Defu Lian, and Zheng Liu. Making text embedders few-shot learners, 2024.
14. Rui Meng, Ye LIU, Shafiq Rayhan JOTY, Caiming XIONG, Yingbo ZHOU, and Semih YAVUZ. Sfr-embedding-mistral: Enhance text retrieval with transfer learning [salesforce ai research blog]. 2024. *Available also from: https://blog. salesf orceairesearch. com/sfr-embedded-mistral.*
15. Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

16. Xuefeng Su, Ru Li, Xiaoli Li, Jeff Z Pan, Hu Zhang, Qinghua Chai, and Xiaoqi Han. A knowledge-guided framework for frame identification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5230–5240, 2021.
17. I Sutskever. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.
18. Fabio Tamburini. Combining electra and adaptive graph encoding for frame identification. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1671–1679, 2022.
19. Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
20. Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. C-pack: Packaged resources to advance general chinese embedding, 2023.
21. Ce Zheng, Xudong Chen, Runxin Xu, and Baobao Chang. A double-graph based framework for frame semantic parsing. *arXiv preprint arXiv:2206.09158*, 2022.
22. Hai-Tao Zheng, Zuotong Xie, Wenqiang Liu, Dongxiao Huang, Bei Wu, and Hong-Gee Kim. Prompt learning with structured semantic knowledge makes pre-trained language models better. *Electronics*, 12(15):3281, 2023.

A Appendix

A.1 Training using exemplars data

Consistent with findings from [2] and [1], our experiments using exemplar data for training—while testing on the same test split as in previous experiments—showed lower performance, as indicated in Table 7. Additionally, when exemplars are used as initial training data (i.e., training with exemplars first and then continuing with the official training split of the two datasets), the performance remains slightly lower compared to not using exemplars at all. We hypothesize that this drop in performance stems from the nature of exemplar data, where each sentence typically evokes only one frame, with annotations limited to a single frame per sentence. This characteristic makes exemplar data less suitable for training models intended to detect multiple frames per sentence, as previously noted by [2].

Table 7. Performance (%) with and without training on exemplars data

Metrics	Accuracy	Precision	Recall
Training with just the training set			
FrameNet 1.5	44	52	44
FrameNet 1.7	48	49	49
Initial training using "exemplars" data then continue training on training set - Test performed using the testset			
FrameNet 1.5	87	89	88
FrameNet 1.7	92	94	95

A.2 Additional results for different recall@k

For further comparison with SOTA approaches, we provide Table 8 that reports performance for different recall @ k. Our method outperform the previous best approaches (KGF1 and COFFTEA)

A.3 Details about the frames detection component

To reduce the complexity of frame detection using a Large Language Model (LLM) as shown in Figure 3, the input comprises a sentence alongside a list of potential candidate frames that could be evoked by the sentence.

Table 8. Performance (%) on FrameNet 1.5 and FrameNet 1.7

Approach	**FrameNet 1.5**			**FrameNet 1.7**		
	Recall@1	**Recall@3**	**Recall@5**	**Recall@1**	**Recall@3**	**Recall@5**
KGFI	86	-	-	86	-	-
COFFTEA	88	93	95	87	93	94
RCIF (Fine-Tuning) [without Info on # Gold Frames]	89	94	95	90	95	96

The following figure shows how our method effectively reduces the complexity of the frame detection task.

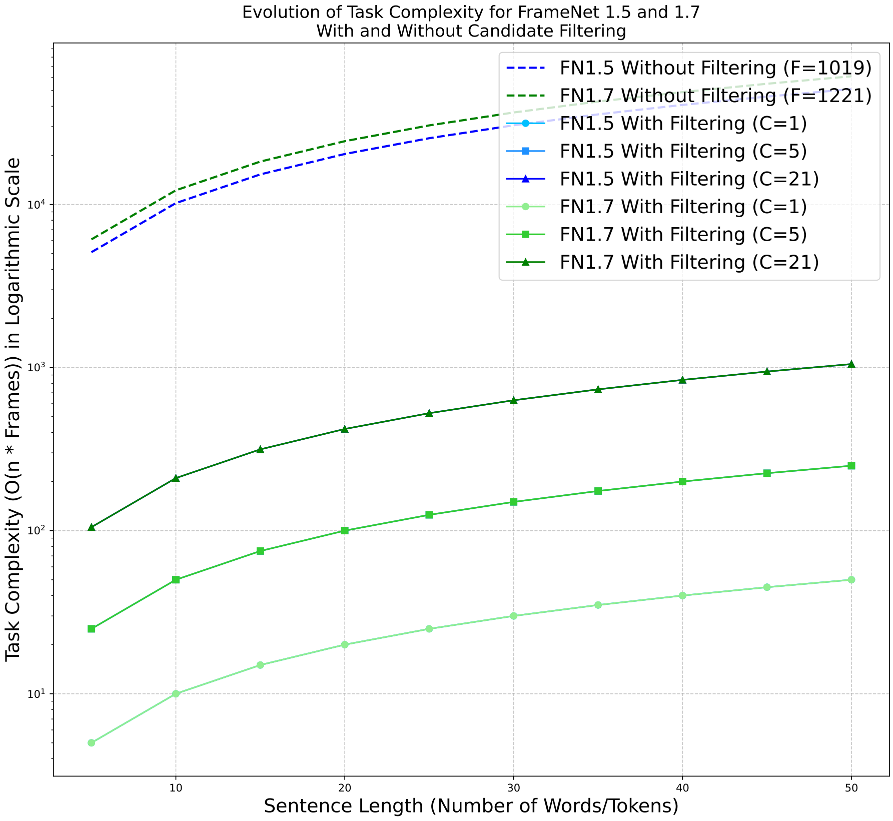


Fig. 3. Complexity evolution of the task of frame detection with and without candidates filtering and different values for the number of candidates *C*.

A.4 Phi-4 and Qwen additional details

For our additional experiments with alternative large language models (LLMs), we employed two advanced models: Phi-4[3] and Qwen2.5-7B-Instruct[4]. Phi-4 is a 14B parameter dense decoder-only Transformer model, designed with a focus on high-quality data and advanced reasoning. It was trained on a blend of synthetic datasets, filtered public domain content, and academic resources, ensuring a strong foundation for instruction adherence and safety. Through supervised fine-tuning and direct preference optimization, Phi-4 achieves precise

[3] https://huggingface.co/microsoft/phi-4

[4] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

alignment and robust instruction-following capabilities. In contrast, Qwen2.5-7B-Instruct is part of the Qwen2.5 model series, featuring enhanced knowledge, stronger coding and mathematical capabilities, and improved instruction following. It generates structured outputs such as JSON, and excels in handling structured data like tables.

Authors

Papa A.K.K. Diallo received the master's degree from the School of Control and Computer Engineering, North China Electric Power University, Beijing, in 2021. He is currently a PhD student at Polytechnique Montreal and is affiliated with Mila - Quebec Institute of Artificial Intelligence. His current research interests include neural machine translation, question answering, language models, knowledge bases, information retrieval and semantic web.

Dr. Amal Zouaq Amal Zouaq is a Full Professor at Polytechnique Montreal in the Computer Science and Software Engineering department. She holds an FRQS (Dual) Chair in AI and Digital Health. She is also affiliated with MILA (Montreal Institute for Learning Algorithms) as an Associate Academic Member. Dr. Zouaq's research interests encompass artificial intelligence, natural language processing, and the Semantic Web. As the director of the LAMA-WeST research lab (http://www.labowest.ca/), her work extends to various facets of natural language processing and artificial intelligence, including LLMs with nonparametric memories, modular LLMs, neuro-symbolic models and the Semantic Web. Furthermore, she actively contributes to the academic community by participating as a program committee member in numerous conferences and journals related to knowledge and data engineering, natural language processing, and the Semantic Web.