Team NLPeers at Chemotimelines 2024: Evaluation of two timeline extraction methods, can generative LLM do it all or is smaller model fine-tuning still relevant ?

Nesrine Bannour[1], Judith Jeyafreeda Andrew[1,2], Marc Vincent[1]
[1]Université de Paris, Imagine Institute, Data Science Platform,
INSERM UMR 1163, F-75015, Paris, France,
[2]PaRis Artificial Intelligence Research InstitutE (PRAIRIE), Paris, France
firstname.lastname[at]institutimagine.org

Abstract

This paper presents our two deep learning-based approaches to participate in subtask 1 of the Chemotimelines 2024 Shared task. The first uses a fine-tuning strategy on a relatively small general domain Masked Language Model (MLM) model, with additional normalization steps obtained using a simple Large Language Model (LLM) prompting technique. The second is an LLM-based approach combining advanced automated prompt search with few-shot in-context learning using the DSPy framework. Our results confirm the continued relevance of the smaller MLM fine-tuned model. It also suggests that the automated few-shot LLM approach can perform close to the fine-tuning-based method without extra LLM normalization and be advantageous under scarce data access conditions. We finally hint at the possibility to choose between lower training examples or lower computing resources requirements when considering both methods.

1 Introduction

The advent of auto-regressive Large Language Models (LLMs) has taken the NLP field by storm and has been diffusing to more specialized domains, such as clinical NLP ever since. While the most powerful models are still only available as private owned services - oftentimes precluding their use with sensitive medical data - open source and open weight models have been catching up, mostly since the release of the LLaMA model family (Touvron et al., 2023). With such open models, in-context learning strategies became more viable. On top of those LLMs, an ecosystem of tools and frameworks has also emerged to provide more robust and efficient ways to use them. One such framework is DSPy (Khattab et al., 2023), whose ambition is to provide a principled and automated way to search LLM prompts and weights and ultimately build robust LLM pipelines.

While this latter technology still evolves, older and more established deep learning models coexist, and the comparative advantages of the two approaches are being assessed. A prominent example of those predecessors is BERT-based models, which can still be considered LLMs, although they are usually an order of magnitudes smaller than their auto-regressive counterparts. With such Masked Language Models (MLMs), fine-tuning of the model weights can be more easily performed due to their usually smaller size.

Temporal Relation Extraction (TRE) is a crucial task for several domains, particularly the clinical domain, requiring a deep understanding of natural language. With the rise of LLMs, recent research efforts attempt to apply these models to the TRE task, but results are still debatable (Han et al., 2023; Chen et al., 2023; Li et al., 2023a). In this paper, we address the clinical event-to-time expression relation extraction task by evaluating two timeline extraction methods.

The main contributions of this paper are:

- An MLM-based fine-tuning approach using a relatively light state-of-the-art MLM model.
- An automated few-shot prompting approach with an LLM using the DSPy framework.
- An evaluation and comparison of these two TRE methods, as well as two temporal expressions normalization methods: a pre-existing tool and a proposed LLM-based method.

2 Related Work

Rule Based Methods. Several research papers (Gaizauskas et al., 2006; Zhou et al., 2008; Hernández et al., 2016; Wang et al., 2016) used rule based approaches for TLINK classification. Zhou et al. (2008) and Hernández et al. (2016) used external clinical domain knowledge to improve the rules.

Machine learning methods. Research efforts

Proceedings of the 6th Clinical Natural Language Processing Workshop, pages 406–416

using Machine Learning approaches for TRE included the use of Support Vector Machine (SVMs) (Lee et al., 2016; Khalifa et al., 2016); Conditional Random Fields (CRFs) (Khalifa et al., 2016); Convolutional Neural Networks (CNNs) (Li and Huang, 2016; Chikka, 2016), and Bi-LSTMs (Tourille et al., 2017a).

Hybrid Approaches. Tang et al. (2013) proposed a hybrid method using a combination of SVM and CRF techniques, with rules to resolve conflicting cases. Nikfarjam et al. (2013) used a SVM with a sentence-level graph-based inference mechanism. Tourille et al. (2017b) used a SVM with word embeddings approach to extract temporal relations.

Language Models. Huguet Cabot and Navigli (2021) presented REBEL, which is a seq2seq model using the BART model as the base model for end-to-end relation extraction. REBEL takes as input raw input and outputs a set of triplets with relations and entities that have been linearized. Eberts and Ulges (2019) used pre-trained BERT as a base model. Entities are detected among all token spans. Entities with no relations are filtered out, and the remaining entities and their relations are classified. Lin et al. (2021) proposed the EntityBERT model obtained with continued pre-training on PubMedBERT base uncased with MIMIC-BIG and MIMIC-SMALL using Entity-Centric masking. The authors then fine-tune EntityBERT for several tasks, including TRE.

Prompt Learning With the increased use of LLMs, prompt Learning has gained popularity. Within this context, several prompting techniques have been proposed using prompt templates (Jiang et al., 2020; Shin et al., 2020; Liu et al., 2023; Li and Liang, 2021; Lester et al., 2021). Few-shot prompting can be used to enable in-context learning, where we provide demonstrations of the prompt to steer the model to better performance. Frameworks such as DSPy (Khattab et al., 2023) allow for optimized few-shot prompting approaches.

3 Task description and data

We participated in the first subtask of the Chemotherapy Treatment Timelines Extraction Shared Task[1] (Yao et al., 2024), which aims to extract temporal relations between chemotherapy events and time expressions and then produce the final patient-level timelines by resolving duplication and conflicts in the pairwise temporal relations. The types of relations to extract are mainly CONTAINS, BEGINS-ON, and ENDS-ON. The data provided by the University of Pittsburgh/UMPC, through a Data Use Agreement during this shared task, includes a list of available de-identified Electronic Health Record (EHR) notes for patients with breast, ovarian, and melanoma cancer. Further details about the subtask and the data distribution are described in Yao et al. (2024). The organizers provide a baseline system based on the Entity-BERT model (Lin et al., 2021).

4 Methods

This section describes our proposed methods for the TRE task and the post-processing, normalization, and summarization steps used to construct patient-level timelines.

4.1 MLM fine-tuning

Model fine-tuning As a core model for the fine-tuning approach, we use `DeBERTa-v3 base` (He et al., 2021), which is a relatively light state-of-the-art 86 million parameters MLM initially trained on 160 GB of general domain text data.

The MLM was fine-tuned on a $(event, time)$ pair multi-class classification task, with processed examples coming from the gold entities and relations dataset provided in the contest training set (which -for the purpose of fine-tuning- was subdivided into a training set and validation set based on which epoch selection was done). The fine-tuned model was then tested on the contest's development set.

The finetuning was done using the huggingface's transformer library using a multiclass classification setup. Given time constraints, a single set of hyperparameters was used for the training and given to the Trainer class of huggingface's library. Learning rate was set to 2e-5 with a weight decay of 0.01, the maximum number of epochs was set to 10 (with an epoch evaluation strategy). The label was one of: $\{begins\_at, ends\_at, contains\_1, no\_link\}$, no_link indicating an absence of a relation between the *event* and *time* entity. After the classification of each candidate pair, the ones predicted to be no_link (i.e. non-existing pairs) were discarded.

Candidates selection The examples themselves were either taken from the list of gold (existing) pairs of related events and time entities or from pairs made of unrelated events and time entities.

[1] https://sites.google.com/view/chemotimelines2024/home

Statistics computed on the training set were used to limit the number of pairs considered: with a crude character count, it was seen that the inner distances between entities involved in temporal relations were never over 213 characters. A threshold of a maximum distance of 300 characters based on that observation was used to limit the number of candidates considered for both the training and classification process. It effectively decreased the number of those candidates to 1/3rd of the possible pairs.

Text pre-processing The text was made of a window centered around the $(event, time)$ entity pair extracted from the full clinical text. Maximum margins of 200 characters before the earliest entity and after the latest entity were taken to add context. As additional pre-processing, the extracted text was modified so the time entity was preceded by a '(TIME=) ' string, while the event entity was preceded by an '(EVENT=)' string. This processing was done in order to signify to the model which terms to look at to classify the provided text based on the candidate pair corresponding to that particular text (and assuming that other pairs might exist in the same text span).

4.2 Automated few-shot prompting with an auto-regressive LLM

DSPy framework DSPy[2] (Khattab et al., 2023) is a framework developed by the Standford NLP group, which aims to optimize LLMs prompts algorithmically. This framework offers two main concepts: Signatures and Teleprompters. The signature is a declarative specification of the input/output behavior of a DSPy module, including a simple description of the task to be solved and descriptions of the input and output fields. Teleprompters are optimizers that can learn to bootstrap and automatically select effective prompts for the program modules. Compiling a DSPy program is based on a training set, a metric to maximize for validation, and a specific teleprompter. DSPy generates new, efficient prompts to match the changes made whenever a code, data, or metric is modified. DSPy also offers several optimizers and advanced features, but due to time constraints, we focused solely on using the BootstrapFewShotWithRandomSearch optimizer while developing our approach. This optimizer self-generates complete demonstrations several times and performs a random search over these generated demonstrations to select the best program.

Automated few-shot prompting As previously stated, we use the DSPy framework to develop and prompt our LLM-based approach. We first convert our input examples into all possible candidate pairs of $(event, time)$ using the gold annotations of entities and relations. For each combination, we extract the corresponding text from the document, which only contains the mentions of these entities. The corresponding text could be a small or a large portion of the full clinical text. Using DSPy, we defined a signature with instructions specifying the three possible types of relations and a description of the expected output format. Then, to cast the TRE task into a generation task, we evaluated these two configurations:

- **Predicting the relation triplet (event, relation, time).** By asking a question with a pair of $(event, time)$ and giving the corresponding text, we prompt our model to predict exactly an ordered list containing the event, the relation type, and the temporal expression. If no relation is found, the model should return an empty list. The basic idea behind this task design is to restrict the model to generate a specified format, avoiding extensive answers and hallucinations. Moreover, this output format is intended to prevent complex postprocessing strategies required to convert expected outputs into valid structures. This is the design we followed for the official submission.
- **Predicting only the relation type.** By asking a question with a pair of $(event, time)$ and giving the corresponding text, similarly to the previous system, we prompt our model to predict solely the relation type between the two entities in the pair. If no relation type is found, an empty list should be returned. This formulation mainly aims to simplify the task to the model. This configuration is evaluated after the official submissions of the shared task.

Figure 1 illustrates our used signatures (prompts) for both configurations. DSPy adds the reasoning statement in the ChainOfThoughts setting, which the LLM will generate to explain the task and the potential steps needed to generate the final answer. This reasoning step generally starts with a general statement, "Let's think step by step in order to *answer the question* or *produce the answer*", followed

[2] https://github.com/stanfordnlp/dspy

by a tailored statement that the model will generate to answer the specific question in the demonstration. For instance, *"we need to find if the chemotherapy event 'carbo' and the date '8/23' have a specific relation. In the text, it is mentioned that ... This indicates that the chemotherapy event began in 8/23"*. Moreover, the automatically selected few-shot examples will be included as in-context demonstrations. As shown in Figure 1, to help the model produce the correct answer, we modified the CONTAINS relation type to CONTAINED-BY, particularly in the first configuration, in which the model must output an ordered list as an answer.

Experimental settings We conducted our experiments using the `Mixtral-8X7B-Instruct-v0.1` language model from Mistral AI (Jiang et al., 2024). We generate up to 256 tokens and set the temperature generation parameter to 0. For both configurations of our automated few-shot prompting LLM approach, we use the BootstrapFewShotWithRandomSearch optimizer to select automatically *k* few-shot examples. These few-shot examples are either chosen from given labeled training data or self-generated based on this data. Indeed, based on the examples in the labeled training data, the DSPy program uses the LLM to produce similar generated few-shot examples. As parameters, we generated 3 candidate programs, kept the maximum labeled examples to the default value, i.e., 16 examples, and set the maximum bootstrapped demos to 4. After converting the shared task training set into possible pairs of $(event, time)$ and the corresponding text, we subdivided this set into a training and a validation set (80/20). The validation set was mainly used to optimize the selection of few-shot examples from the training set[3].

4.3 Normalization and patient-level summarization

The triplets of relations *(event, relation_type, time)* obtained in earlier steps had their time mention processed -if necessary- to produce a normalized $TIMEX3$ expression in the form of a date. Two methods were used in order to do so: Heideltime and a simple LLM-based query with hand-made few shot examples. Both methods could take as input the time expression of the considered $(event, time)$ pair, but also -if present for the document containing the pair- the $document\_creation\_time$ (in the form of a date).

HeidelTime normalization As a first method to normalize the temporal expressions, we use a Python wrapper for the HeidelTime tool (Strötgen and Gertz, 2013), namely *py_heildetime*[4]. HeidelTime extracts and normalizes temporal expressions according to the TIMEX3 standard. The relative temporal expressions are normalized using the $document\_creation\_time$ (DCT). Since HeidelTime did not normalize relative temporal expressions such as *currently*, we normalize it to the DCT. This method was applied to both the outputs of the LLM-based TRE approach and the MLM fine-tuning approach.

LLM-based query normalization This second method was used only for the MLM fine-tuning approach of the official submission. The latest state-of-the-art 7 billion parameters, `OpenChat 3.5` model (Wang et al., 2023a), was used through a local serving of an openai compatible API. The request itself was made of three parts.

prompt part 1 was used everytime:

> *"please normalise the following string to a date format YYYY-MM-DD or, if you can't to a YYYY-MM format*"

prompt part 2 was appended to $prompt1$ if a document time was available (with <*doc_time_input*>, a place holder to be replaced with the document date string:

> *"(the time at which the document is redacted is <doc_time_input>)"*

prompt part 3 was used everytime, giving the time expression to normalize. It was appended to the previous part:

> *": <time expression>"*

From the former prompt and 6 short hand-picked synthetic examples in the form of triplets *(time_expression, doctime or None, answer_date or error_string)*, a few shot strategy was implemented as a user/assistant dialog.

Summarization To provide a timeline from the triplets obtained earlier, summarization was performed as follows. First, we discarded the triplets containing a *time* mention not matching the Python regular expression:

[3] More details about the DSPy implementation code can be found in Khattab et al. (2023) and `https://github.com/stanfordnlp/dspy`

[4] `https://github.com/hmosousa/py_heideltime`

Respond to the question based on the given text.
The possible answers are: 'CONTAINED-BY', 'BEGINS-ON', 'ENDS-ON'.

Follow the following format.

Question: ${question}

Text: ${text}

Reasoning: Let's think step by step in order to ${produce the answer}. We ...

Answer: a list containing only the relation. If no relation is found, the answer is solely an empty list.

Question: Given this chemotherapy event: ${EVENT} and this temporal expression: ${TIMEX}, which is the relation between these entities, if any?

Text: ${text}

(a) Prompting the model to output the $relation\ type$ between the given $(event, time)$ pair.

Respond to the question based on the given text.
The possible answers are: 'CONTAINED-BY', 'BEGINS-ON', 'ENDS-ON'.

Follow the following format

Question: ${question}

Text: ${text}

Reasoning: Let's think step by step in order to ${produce the answer}. We ...

Answer: Each answer is an ordered list, containing the chemotherapy event, then the corresponding answer then the temporal expression. If no relation is found, the answer is an empty list.

Question: Given this chemotherapy event: ${EVENT} and this temporal expression: ${TIMEX}, which is the relation between these entities, if any ?

Text: ${text}

(b) Prompting the model to output the relation triplet $(event, relation, time)$ given the $(event, time)$ pair.

Figure 1: The two defined DSPy signatures to prompt our automated few-shot prompting LLM approach.

`'^([0-9]{4})-([0-9]{2})-([0-9]{2})$'`

Then, following the organizers' instructions, for groups of triplets sharing the same date and event but with different relation types, i.e., *contains-1* and a more precise type (*begins-on*, *ends-on*), only the more precise mentions were kept. At last, triplets were de-duplicated and sorted.

5 Evaluation metrics

For the final evaluation of patient timelines, the organizers provide an evaluation code[5]. The evaluation process covers strict and relaxed evaluation settings by calculating the average F1 score across all patients. The official score is an arithmetic mean of two types of Macro F1 measure, type A and type B, in a relaxed to-month setting. The type A evaluation includes the patients with no gold timelines, while the type B evaluation excludes the patients with no true relations. The relaxed to-month setting means only the month must match the gold annotation. More details about the evaluation process are presented in the shared task website[6]. While selecting the different models we tried, we evaluated them based on the official score provided by the organizers.

For the optimization of our automated few-shot LLM approach and to ensure quality few-shot examples and demos, we defined a strict F1 measure. Indeed, the DSPy optimizer will only keep the few-shot examples that maximize this evaluation metric. Note that for our first configuration setting, i.e., predicting an ordered list of $(event, relation, time)$, the system prediction will not be considered a match if the model correctly predicts the relation type but fails to output the required format.

6 Results & discussion

To participate in this shared task, we submitted two runs. The first run is the MLM fine-tuning approach (**NLPeers 1**), and the second run is the automated few-shot prompting LLM approach (**NLPeers 2**). In this section, we begin by discussing the overall performance of our systems on the test and the development sets. Since the gold annotations of the test set will not be released, we then present a more in-depth review of each of our methods on this set and the impact of adding the LLM-based

[5] https://github.com/HealthNLPorg/chemoTimelinesEval

[6] https://sites.google.com/view/chemotimelines2024/evaluation

normalization on performance.

6.1 Overall performance

Table 1 presents the official subtask 1 results at the patient-level of our submitted methods and the organizers' baseline system[7] on the test set, including average scores and scores per cancer type, as reported in the Leader board of the Chemotimelines shared task. The MLM fine-tuning approach outperforms the automated few-shot LLM-based approach on the test set, with an average score of 0.77 vs. 0.64. However, the best results are obtained with the baseline system, with an average score of 0.89.

In comparison with submissions from other participants for this subtask, we are the third-best team, out of eight teams, in terms of average score if we consider our submission of MLM fine-tuning approach (**NLPeers 1**). It's worth noting that only the top team outperformed the baseline system. On the Melanoma dataset, we are in second place with a score of 0.84 vs. 0.87 for both the top team and the baseline system).

Table 2 summarizes the results of our proposed approaches on the development set, including the additional experiments of LLM-based query normalization and predicting the relation type for the few-shot prompting LLM approach, which were not part of our official submissions. Similar to the results on the test set, the fine-tuned MLM approach (an average score of 0.85) outperforms the automated few-shot LLM approach in both configurations using the HeidelTime normalization (an average score of 0.61 for the relation type prediction and 0.56 for the relation triplet prediction). However, using both HeidelTime and LLM-based query normalization enhanced the results of the relation triplet prediction, hinting at the fact that performances measured on the test set could probably have been higher if the combined normalization was applied to the automated few-shot LLM approach. Interestingly, the official submission models have performances that vary in opposite directions when going from the development set to the test set: the fine-tuned MLM model performance decreases while the few-shot one increases.

6.2 Performance of fine-tuning MLM model

Whole set relation type errors Table 3 represents a confusion matrix computed on the development set after applying the fine-tuned MLM model, it compares the gold relation types to the predictions made. As can be seen in this table, a major source of error on the development set for this method is the mislabeling of 'false' (*no_link*) candidate triplets as *contains* triplets. It accounts for roughly 10% of the *no_link* candidates. This is not unexpected since *no_link* candidates are by far the first class present in the used development set (655 total, after filtering based on entity distance), followed by 'contains' triplets which represent roughly half of the former ones (354 total).

Next error based on absolute count are *ends-on* relations mislabeled as *begins-on* (38/83), while the converse almost never occurs (2/103), although the categories *begins-on* and *ends-on* are almost balanced (respectively 103 and 83 occurrences).

Melanoma subset relation type errors As out of the of the three cancer subsets, the model seemed to perform relatively worse on the melanoma, we inspected further the errors specifically made for the melanoma subset, it appears that it responsible for the vast majority of the *no_link* candidates mislabeled as *contains* triplets made in the general set (i.e. 64 out of the 68 counted in Table 3). Interestingly, the melanoma subset also accounts for 41 out of the 49 *begins-on* relations mislabeled as *contains*. This relative concentration of errors in the melanoma subset could be explained by the lower count of melanoma examples in the training set, increasing the odds that the model learned undue associations specific to that subset.

6.3 Performance of automated few-shot LLM prompting

As reported in Table 2, using the HeidelTime normalization, predicting relation type yields better results than predicting relation triplets, with an average F-measure of 0.61 vs. 0.56. This could be due to the strict evaluation of the triplet configuration. Indeed, as already mentioned, no extra post-processing steps are taken for the outputs. Results per cancer are jointly discussed, along with the impact of normalization methods, in the next section.

Among all the possible candidate pairs (1287), the **relation triplet** model predicts 1046 tuples and 241 empty lists. Among the 1046 tuples, 133 are invalid, i.e., not corresponding to an ordered list $(event, relation, time)$ or not mentioning the correct $event$ or $time$ present in the input. Among

[7] https://github.com/HealthNLPorg/chemoTimelinesBaselineSystem

Approach	Average Score	Breast cancer	Melanoma	Ovarian
Fine-tuned MLM + HeidelTime & OC normalization *(NLPeers 1)*	0.77	0.72	0.84	0.75
Automated few-shot LLM **(Relation triplet)** + HeidelTime normalization *(NLPeers 2)*	0.64	0.49	0.81	0.63
Baseline system	0.89	0.93	0.87	0.88

Table 1: The official results on the test set. OC refers to the LLM-based normalization using the `OpenChat` model.

Approach	Average Score	Breast	Melanoma	Ovarian
Fine-tuned MLM				
Relation type (classification) + HeidelTime & OC normalization *(official submission, NLPeers 1)*	0.85	0.84	0.81	0.88
Relation type (classification) + HeidelTime normalization *(non official submission)*	0.74	0.61	0.85	0.76
Automated few-shot LLM				
Relation triplet (generation) + HeidelTime & OC normalization *(non official submission)*	0.72	0.70	0.74	0.71
Relation type (generation) + HeidelTime normalization *(non official submission)*	0.61	0.57	0.78	0.48
Relation triplet (generation) + HeidelTime normalization *(official submission, NLPeers 2)*	0.56	0.53	0.70	0.47

Table 2: The results on the development set. OC refers to the LLM-based normalization using the `OpenChat` model.

the remaining 913 valid tuples, 146 are correct (69 *begins-on*, 34 *ends-on*, 43 *contains*). As for the **relation type** model, among all the possible candidate pairs (1287), it predicts 994 relation types and 293 empty relations. Among the 994 predicted relation types, 824 respect the expected output format, and 154 are correct (90 begins-on, 47 ends-on, 19 contains).

Table 4 presents the number of semantic errors, as defined in Li et al. (2023b), as well as some semantically incorrect samples on the development set for both configurations of automated few-shot LLM approach, using the HeidelTime normalization. A semantic error is defined as a relation type that does not exist in the pre-defined set of relation types. Looking at this table, we notice that although the relation triplet prediction model produces a total of 43 errors, only 7 types of errors are generated and seem semantically "correct" but are out of the pre-defined relation type set. However, the relation type prediction model produces only a total of 16 errors, including 11 different types of relation types, which seems less precise. Indeed, the relation type prediction model tends to generate large texts containing not only the relation but also explanations and hallucinations. Though the main idea behind relation type prediction is to simplify the relation extraction task to the LLM, we believe that reformulating the task with structured instructions and input/output examples, such as our triplet prediction method, could provide better results, using the appropriate pre- and post-processing steps, as already stated in previous research works (Li et al., 2023b; Lu et al., 2022; Wang et al., 2023b).

Gold	BEGINS-ON	CONTAINS	ENDS-ON	no_link
BEGINS-ON	52	18	30	0
CONTAINS	49	328	30	68
ENDS-ON	2	1	11	0
no_link	0	7	12	587
Total	103	354	83	655

Table 3: Confusion matrix for the MLM fine-tuning approach applied on the development set.

	Relation Type Error	Semantically incorrect samples
Automated few-shot LLM **Relation triplet** + Heideltime *(official submission, NLPeers 2)*	43	occurs on, occurs-on, contained-in, not going to occur, not related, duration, ended-on
Automated few-shot LLM **Relation type** + Heideltime *(non official submission)*	16	answer, be, beg, begins, conta, during, every-on, happening-on, happens-on, lasts-for, planned-for

Table 4: Semantic errors and semantically incorrect samples on the development set.

6.4 Impact of LLM normalization on performance

It should be noted here that although we used both Heideltime and an LLM-based normalization for the official MLM fine-tuning results, due to time constraints, only the Heideltime normalization was made available for the official automated few-shot prompting results. A comparison of results with and without said LLM normalization was done after the official results on the development set. The results in Table 2 show that the additional Open Chat normalization has a strong impact on both predictors, with an increase ranging from 11 (fine-tuned MLM) to 16 points (automated few-shot prompting) on average score. This seems to suggest that such a simple prompt method can be efficient for this kind of task, where a very limited context and no specific background knowledge is needed to answer the query at hand, thus requiring no complex task description or prompt search strategy.

A more detailed look at the impact of the complementary normalization per cancer type seems to indicate that breast and ovarian subsets benefit the most regardless of the model. A detailed inspection of the differences in time expression patterns highlights that the temporal expressions of melanoma are less varied, with different pattern proportions. For example, in the few-shot LLM triplet prediction, *currently* - which is well normalized by Heideltime when the document time is given - accounts for 20% of temporal expressions of melanoma, but only 5% in other cancer. In the same way, *today* accounts for 35% of melanoma time expressions and 28% of other cancers time expressions.

While we tested two normalization approaches, the reference one provided as a scala library by the contest organizers was not tested as we failed to include it in time in our otherwise Python-based code. The effect on the measured performances (and comparison to other teams' proposals using it) is difficult to assess as - besides the respective merits of each method - the reference time normalization was used as a gold standard for the evaluation process. In effect, terms that it could not normalize were discarded, transforming potentially correct time expressions and relations to perceived incorrect ones.

7 Conclusion

In this work, we showed that an LLM-based automated prompting method could, with no weight fine-tuning, give good results on a temporal relation extraction task. We also showed that a smaller fine-tuned MLM likely performs better while requiring less computing resources, thus confirming that smaller model fine-tuning is still relevant for such classification tasks. Given the low number of examples retained at the end of the selection procedure by the few-shot prompting approach, it can

be inferred that a smaller set of examples could be used to reach better performances, effectively making it an interesting choice when access to annotated data is scarce. Another finding demonstrated the effective use of a simple LLM approach for a general domain task such as time normalization.

Limitations

It is to be noted that the time devoted to developing both proposed methods was limited due to late enrollment in the shared task and access to data. The methods were also mostly developed from scratch w.r.t. the timeline prediction perspective. This strongly suggests that both approaches could be improved. As such, more work is warranted to get these proposed solutions closer to being the best-performing ones. For the automated few-shot prompting LLM solution, creating a true pipeline chaining multiple steps (e.g., verification/enrichment) could greatly increase the accuracy of the provided answers. Indeed, more evaluation steps should be included, in particular for the clinical domain, to avoid inaccuracies in the generated reasoning steps and demonstrations. Another improvement would be to use rule-based post-processing steps to deal with the inherent variability of answers produced by the LLM. Further research into using DSPy, particularly its advanced prompting and optimization features, could also be conducted. For the fined-tuned MLM approach, proper parameter selection could increase the performance and stability of the model. On a last note on the two proposed methods, we considered them as exclusive to one another to measure their respective benefits, but a combination of both could allow the final result to get even better performances. Finally, we did not compare our normalization process to the one provided as a gold standard, making it more difficult to draw definitive conclusions based on the final evaluation of our proposal and its comparison to other participants performances.

Ethics statement

Using Large Language Models (LLMs) in the clinical domain raises several ethical concerns. First, due to the sensitive nature of clinical data, special precautions must be taken while working with it. This work uses de-identified clinical data obtained through a Data Use Agreement. Therefore, the designed prompts for our LLM-based methods do not contain identifying personal information about patients. Second, a major challenge while leveraging LLMs, particularly in clinical research, is the transparency and interoperability of results. Indeed, these models often act as 'black boxes,' making it hard to understand the generated outputs and the decisions made, which is crucial for clinicians. As a result, a human and expert evaluation is required, first for minimizing hallucinations, biases, and harmfulness outputs and then for evaluating and validating the coherence of generation. Third, LLMs are complex models with billions of parameters that necessitate lots of computational resources, thus generating a carbon footprint. This is also valid for fine-tuning the MLMs-based models. Finally, it is worth noting that the proposed methods are mainly for research purposes, and additional studies need to be conducted before integrating them into practical applications, where the goal is to help clinicians conduct a systematic analysis of large patient records.

Acknowledgments

This work was supported by state funding from the "Agence Nationale de la Recherche" under "Investissements d'Avenir" programs Institut Hospitalo-Universitaire Imagine (ANR-10-IAHU-01), CDE.AI (ANR-21-PMRB-0002), PRAIRIE 3IA Institute (ANR-19-P3IA-0001).

References

Qingyu Chen, Jingcheng Du, Yan Hu, Vipina Kuttichi Keloth, Xueqing Peng, Kalpana Raja, Rui Zhang, Zhiyong Lu, and Hua Xu. 2023. Large language models in biomedical natural language processing: benchmarks, baselines, and recommendations. *arXiv preprint arXiv:2305.16326*.

Veera Raghavendra Chikka. 2016. CDE-IIITH at SemEval-2016 task 12: Extraction of temporal information from clinical documents using machine learning techniques. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1237–1240, San Diego, California. Association for Computational Linguistics.

Markus Eberts and Adrian Ulges. 2019. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*.

R. Gaizauskas, H. Harkema, M. Hepple, and A. Setzer. 2006. Task-oriented extraction of temporal information: The case of clinical narratives. In *Thirteenth International Symposium on Temporal Representation and Reasoning (TIME'06)*, pages 188–195.

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors. *arXiv preprint arXiv:2305.14450*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *Preprint*, arXiv:2111.09543.

Eddie Paul Hernández, Alexandra Pomares Quimbaya, and Oscar Mauricio Muñoz. 2016. Htl model: A model for extracting and visualizing medical events from narrative text in electronic health records. In *ICT4AgeingWell*, pages 107–114.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2370–2381, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Abdulrahman Khalifa, Sumithra Velupillai, and Stephane Meystre. 2016. UtahBMI at SemEval-2016 task 12: Extracting temporal information from clinical text. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1256–1262, San Diego, California. Association for Computational Linguistics.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2023. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*.

Hee-Jin Lee, Hua Xu, Jingqi Wang, Yaoyun Zhang, Sungrim Moon, Jun Xu, and Yonghui Wu. 2016. UTHealth at SemEval-2016 task 12: an end-to-end system for temporal information extraction from clinical notes. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1292–1297, San Diego, California. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023a. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633*.

Peng Li and Heng Huang. 2016. UTA DLNLP at SemEval-2016 task 12: Deep learning based natural language processing system for clinical information identification from clinical notes and pathology reports. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1268–1273, San Diego, California. Association for Computational Linguistics.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023b. CodeIE: Large code generation models are better few-shot information extractors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Toronto, Canada. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2021. EntityBERT: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201, Online. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *AI Open*.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

Azadeh Nikfarjam, Ehsan Emadzadeh, and Graciela Gonzalez. 2013. Towards generating a patient's timeline: Extracting temporal relationships from clinical notes. *Journal of Biomedical Informatics*, 46:S40–S47. Supplement: 2012 i2b2 NLP Challenge on Temporal Relations in Clinical Data.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation*, 47:269–298.

Buzhou Tang, Yonghui Wu, Min Jiang, Yukun Chen, Joshua C Denny, and Hua Xu. 2013. A hybrid system for temporal information extraction from clinical text. *Journal of the American Medical Informatics Association*, 20(5):828–835.

Julien Tourille, Olivier Ferret, Aurélie Névéol, and Xavier Tannier. 2017a. Neural architecture for temporal relation extraction: A Bi-LSTM approach for detecting narrative containers. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 224–230, Vancouver, Canada. Association for Computational Linguistics.

Julien Tourille, Olivier Ferret, Xavier Tannier, and Aurélie Névéol. 2017b. Temporal information extraction from clinical text. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 739–745, Valencia, Spain. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2023a. Openchat: Advancing open-source language models with mixed-quality data. *arXiv preprint arXiv:2309.11235*.

Wei Wang, Kory Kreimeyer, Emily Jane Woo, Robert Ball, Matthew Foster, Abhishek Pandey, John Scott, and Taxiarchis Botsis. 2016. A new algorithmic approach for the extraction of temporal associations from clinical narratives with an application to medical product safety surveillance reports. *Journal of Biomedical Informatics*, 62:78–89.

Xingyao Wang, Sha Li, and Heng Ji. 2023b. Code4Struct: Code generation for few-shot event structure prediction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3640–3663, Toronto, Canada. Association for Computational Linguistics.

Jiarui Yao, Harry Hochheiser, WonJin Yoon, Eli Goldner, and Guergana Savova. 2024. Shared task: Chemotherapy treatment timeline extraction. *Clinical NLP Workshop, NAACL 2024. Mexico City, Mexico.*

Li Zhou, Simon Parsons, and George Hripcsak. 2008. The Evaluation of a Temporal Reasoning System in Processing Clinical Discharge Summaries. *Journal of the American Medical Informatics Association*, 15(1):99–106.