Exploring Generative approaches for NER on biomedical data

Javier Alejandro Lopetegui González
Université Paris-Saclay
M1-AI
Supervisors: Cyril Grouin, Nona Naderi, Pierre Zweigenbaum
LISN

April 23, 2024

Abstract

Named Entity Recognition (NER) is a broadly studied classification task in Natural Language Processing (NLP). This task is particularly challenging in some contexts, such as biomedical data, due to the heterogeneous nature of this kind of data with very specific concepts and many entities to identify. Several methods have been explored for NER on biomedical data, but there is still much to do. Generative approaches that have proven to be significantly good for NER on several types of data have not shown convincing results on biomedical data. The aim of this work is to explore different ways to address these problems using a generative approach. Particularly, we compare the results of three different approaches: a Mask Language Model (MLM) as a baseline, a Generative model, and a combination of MLM and Generative model.

1 Introduction

Named Entity Recognition (NER) is a classification task in Natural Language Processing that deals with the identification and classification of named items in unstructured data [1]. These items belong to predefined semantic types such as persons, locations, and organizations [2].

NER systems are commonly used as the first step in question answering models [3], machine translation [4], co-reference resolution [5], or information extraction [6][7]. For this work, we are going to focus on the case of NER for information extraction, particularly in the domain of biomedical data.

We have two main motivations to focus on this kind of data. On the one hand, mainly after the Covid-19 pandemic, there has been a significant increase in the availability of biomedical documents in the form of research papers, case reports, electronic health records, and clinical notes [8]. To keep up with the increasing demand for biomedical knowledge, large-scale data management is necessary. In its current state, it is very challenging for researchers to manage and infer information from unstructured texts. On the other hand, biomedical data represents a challenge for NER compared with more common contexts where the current approaches have proven to be sufficiently good. The difficulty of the concepts biomedical data deals with alongside the fact that usually the systems used for medical purposes require an important degree of confidence make it a big challenge of this topic.

Common entities in biomedical data are genes, diseases, species, or chemicals [9]. In a more clinical domain, we have entities such as drugs, conditions, devices, and others [10]. These kinds of entities tend to be very specific for each work, and furthermore, the field is continuously evolving, so there is a need for new and robust approaches to address the task.

1.1 Common approaches for NER on biomedical data

Several techniques have been used for NER on biomedical data. The first approaches correspond to knowledge-based methods. These methods rely on rules, regular expressions, or domain-specific dictionaries to identify entities in texts [11]. The performance of these approaches is limited and very dependent on the specific case of use, even when multiple creative solutions have been found to make them more robust.

Furthermore, machine learning (ML) techniques such as Support Vector Machine (SVM) [12], Hidden Markov Models (HMM) [13], or Conditional Random Fields (CRF) [14] have also been used for NER and have shown very good results with the appropriate feature engineering process [15][16][17].

The emergence of Deep Learning (DL) methods was a breakthrough for NER. It opens the possibility

of learning more complex relations between features automatically. These models are less domain-specific dependent and more robust. Several DL architectures such as LSTM [18], RNN [19], or Bi-LSTM [20] have been widely used.

However, as in many NLP tasks, the development of transformer [21] architectures revolutionized the state of the art in NER. This architecture lets us take advantage of its efficiency to train models on very large data, something impossible before. Then, by just adding some adaptive layers, we can fine-tune models for specific downstream tasks and take advantage of all the knowledge the model has seen during the pre-training process. Models such as **BERT** [22], which stands for Bidirectional Encoder Representations from Transformers, have proven to have very good performance on NER. These kinds of models are known as Masked Language Models (MLM) because of the approach used during training: prediction of original tokens by randomly changing them to a special **[MASK]** token. Even though the current results are really promising, there is still a gap between the results obtained on general data and biomedical data [23].

Currently, with the boom of Generative models after the **Chat-GPT** revolution, several studies have been conducted on NER with generative approaches [24]. This kind of models has shown good results on Zero-shot and Few-Shot learning scenarios for NER on general entities, which is not the case for biomedical data [23][25]. There is still not a clear way to address the problem from a generative perspective. The kind of prompts to use is very dependent on the specific model you use, and even for the same model, there is no clarity about the best way to work.

With that in mind, in the present work, we explore different options to address NER on biomedical data using generative approaches. Firstly, we study two different prompt settings to solve the task with a Few-shot approach. Then we propose a combination of MLM and generative approach as a way to leverage the capabilities of both. We perform experiments on a dataset known as **CHIA** [10], consisting of clinical trial eligibility criteria texts. Following the study by Tian et al. [26], we use **RoBERTa** [27] as baseline for our work.

2 Methods

The aim of this work is to compare three different approaches for NER on biomedical data:

- MLM: Use a MLM as baseline, based on the study by Tian et al [26]
- Generative model: Use a Few-shot approach for NER with a generative model
- MLM + Generative model: Combine the baseline MLM with a generative model to enhance the annotations

In the following three sections we explain each of these approaches.

2.1 MLM

The first approach we work with as a baseline is an MLM. Particularly, we use **RoBERTa** [27] following the fact that it has proven to be effective in the dataset that we use in our experiments [26].

This model is a version of **BERT** model with some training improvements. This model was trained for more epochs and with bigger batches as well as additional data. The next sentence prediction objective was removed from the original **BERT** training setting. They also used longer sequences for training, enhancing the long-term robustness of the model, as well as dynamically updating for masking pattern.

As stated before, MLMs are well-known for their ability to adapt well to different downstream tasks such as NER by just adding customized layers. Particularly, we use a token classification version of **RoBERTa**. It is the model with a token classification head on top (a linear layer on top of the hidden-states output).

2.2 Generative model

For the second approach, we propose the use of a Generative model with a prompt-based method for NER. We explore two different settings for the prompt used in the generation phase with a *Few-shot* approach. In both cases, we add at the beginning of the prompt a brief explanation about the task with the list of entities we are considering, which is a common procedure in current studies [24]. Then we provide one example for the first prompt and two for the second one. In both cases, the examples are very simple, with the goal of illustrating to the model the annotation scheme more than explaining the nature of the entities in the examples. The objective is to evaluate the model's intrinsic capability to solve the task without more information than its own training data.

The main difference between both prompts is in the format of the annotations. In both cases, the input is a plain text sentence. For the first case, the model is expected to annotate in the exact format the data is originally annotated. For the second

case, the output is expected to be the same sentence tagged in place in a format: <entity_name> text corresponding to the entity </entity_name>. In the figures 1 and 2 of Appendix A, we show one example for each of them.

For this work we decided to use **Mistral-7b** model [28]. This is an open-source Large Languege Model (LLM) with seven billions of parameters. Even though it is not as big as other LLMs, it has shown state-of-the-art results on several NLP tasks.

2.3 MLM + Generative model

The last proposal consists of a combination of the first two approaches. The proposed model is an ensemble of an MLM and a generative model. We fine-tune the MLM just as in the first approach. Then, the final annotations are the results of a sort of *verification process* using the generative model.

The idea is to take the annotations of the MLM and create a prompt to put the generative model to verify the annotations and correct them if needed. The final output is the sentence annotated in the format of the second prompt we discussed in the previous section.

The hypothesis behind this idea is that for the generative model, it should be easier to handle a previously annotated sentence and just correct it than making the entire annotation. We consider that this approach can help the model improve the performance on the less represented entities particularly and also make the annotations more precise in general.

In Figure 3 of Appendix A, we show one example of the verification prompt that we define.

3 Experiments and results

In this section, we present the experiments and results using each of the previous approaches for NER on clinical trial eligibility criteria texts. We begin by examining the dataset used for experiments in section 3.1. Then, in section 3.2, we explain the evaluation metrics used in the experiments. Finally, in section 3.3 we show the results and conduct a critical analysis of them, as well as a brief discussion about potential solutions to improve the results.

3.1 Dataset

Following the study by Tian et al. [26][1], we use in this work the **CHIA** dataset [10]. This is a large annotated corpus of patient eligibility criteria extracted from 1,000 interventional, Phase IV clinical trials registered in ClinicalTrials.gov. This dataset includes 12,409 annotated eligibility criteria, represented by 41,487 distinctive entities of 15 entity types and 25,017 relationships of 12 relationship types. The annotations are originally in Brat format.

To maintain consistency with the study we are using as a baseline, we do not consider relationship types or overlapping entities. Furthermore, we converted the original annotations into IOB2 format [2] using the code provided by the authors of the mentioned paper.

The entities selected in the previous study are *Condition, Value, Drug, Procedure, Measurement, Temporal, Observation, Person, Mood, Device, Pregnancy_considerations*. Among these entities, we decided to exclude for our work the entities *Mood* and *Pregnancy_consideration* as they are underrepresented compared with the rest and present conflicts due to the original overlapping entities in the dataset.

For splitting the dataset files into training and test we used the code of the original dataset, to achieve a distribution as close as possible to their distribution. This distribution consists of 1800 training files and 200 test files, totaling 11,102 sentences for training and 1307 for testing. Furthermore, we split the training sentences into train and validation, taking the 20% for validation resulting in 8881 for training and 2221 for validation. The final distribution of entities in the dataset is shown in the Table 1. The resulting version of the dataset used for experiments is available in huggingface [3] to allow the replication of our experiments.

3.2 Evaluation metrics

The evaluation strategy we follow is the same as the one used in the baseline paper to report the results. We use two criteria for evaluation: *strict* and *relaxed*.

For the *strict* mode the true entity and the predicted entity must match completely. It means that the boundaries should be the same as well as the B or I annotation inside the entity. As the format used is IOB2, every entity is forced to start with *B-entity* and not a *B-entity* is allowed inside its boundaries.

On the other hand, for the *relaxed* mode they use a very permissive approach, considering two entities matching if they have at least one token matching in the annotation. For example in the next example we can see two annotations corresponding to the same

[1] link to the github repository with original code for the baseline paper: Clinical-trial-eligibility-criteria-NER

[2] wikipedia link for IOB format explanation: IOB_wiki

[3] dataset

Subset	Condition	Value	Drug	Measurement	Procedure	Temporal	Observation	Person	Device
train	8031	2677	2479	2292	2285	2129	1171	1136	264
val	1913	628	593	534	602	510	328	257	59
test	1104	345	443	288	311	295	166	135	23

Table 1: Entities distribution in our experiments dataset.

sentence where the true entity and predicted entity do not match in *strict* mode but they do it in *relaxed* mode:

- True annotation: *O B-Condition I-Condition I-Condition O*
- Predicted annotation: *O O O I-Condition O*

Finally, we compute the *recall, precision* and *f1-score* in both modes for each entity and overall. We also report a world level overall *accuracy*, which means the accuracy considering each individual annotation, not the full entity.

3.3 Experiments descriptions and results

Now we are going to explain the experiments we use to evaluate the proposed methods. Firstly, we explain the process for selecting the best prompt to use for the generative approach and subsequently the final experiments for each of the three approaches.

3.3.1 Prompt selection for Generative model approach

We conducted a preliminary experiment to select the most promising prompt among the two we proposed. For this experiment, we used a small subset of the testing data. Specifically, we took 50 sentences from the test data and generated the annotations with each prompt.

As we stated before, we use **Mistral-7b** [28] model for generation. In order to save computational power we used a *4-bit* quantization of the model. Previous studies have shown that working with that resolution does not significantly decrease the performance of the model. Furthermore, we use the **text generation pipeline**[4] from huggingface. The exact details of the experiment can be found on the Github repository of this project [5].

Finally, after making the annotations for each prompt in the selected sentences, we perform some processing to align the model output with the original sequence. To do so, we use the same tokenizer to process the outputs and the original sentences, keeping the annotations. Even after that process, we encounter some alignment problems. These problems are mostly related to punctuation signs. In some cases there are unexpected signs at the end of the original sentence which are not replicated by the generative model, while in other cases the model add signs not in the sentence provided as input. We provide two examples to illustrate this:

- Example 1:
  - true tokens: 'severe', 'respiratory', 'disease', ';'
  - true annotations: *B-Condition, I-Condition, I-Condition, O*
  - predicted tokens: 'severe', 'respiratory', 'disease'
  - predicted annotations: *B-Condition, I-Condition, I-Condition*
- Example 2:
  - true tokens: 'Invasive', 'fungal', 'infections', 'in', 'history', 'and', 'at', 'present'
  - true annotations: *B-Condition, I-Condition, I-Condition, O, O, O, O, O*
  - predicted tokens: 'Invasive', 'fungal', 'infections', 'in', 'history', 'and', 'at', 'present', '.'
  - predicted annotations: *B-Condition, I-Condition, I-Condition, O, B-Condition, O, O, O, B-Condition*

As we can see, in both cases, there is a difference of 1 token between the prediction and the ground truth. In the first case, there is a ';' at the end of the original sentence that is not replicated by the model in the output. Then, in the second case, the model adds a period at the end of the prediction that is not in the original sentence. These problems are related with inconsistency in the way of ending the annotations in the dataset, it would be better to standardize in order to have always the same behavior.

Therefore, we only use for the final evaluation those sentences resulting in the same length after

[4]text generation pipeline huggingface: pipelines

[5]Github repository for this work: NER-ClinicalTrials-Elegibility-Criteria

tokenization. For the two prompts, we remove 28% of the sentences. This resulted in the following distribution of true entities for evaluation: total = 292, drug = 17, condition = 234, measurement = 8, value = 7, procedure = 11, temporal = 11, observation = 3, and person = 1.

We can see the experiments results obtained for *strict* evaluation and accuracy over the two prompts in Table 2. We can see that the second prompt shows better results for all the metrics with not significant difference for recall. Taking into account these results we consider that the second prompt is more promising. Thus, we use it as the final prompt configuration in the Generative approach.

		Strict		
Prompt	acc	recall	prec	f1-score
1	0.5583	0.5862	0.4392	0.5022
2	**0.6897**	**0.5958**	**0.6281**	**0.6115**

Table 2: *Strict* criteria metrics and accuracy for the two prompts.

In order to explain the difference in performance we should analyse the difference between the prompts. The first one is that the first prompt just contains one example for the task while the second has two. Anyway, as the examples are not very representative for all the entities and are used just to explain the structure of the model output, we consider that this is not the main source of the difference. Then, the main difference between the two prompts may be due to the fact that in the first one we ask the model to put the output in IOB format and it can add an unnecessary complexity to the task, as it is something that can be easily done as a post-processing step. Furthermore, previous studies suggest that using the output format of the second prompt is a better way to address the task [23].

3.3.2 MLM experiments

For the MLM approach as stated before, we use the huggingface **RoBERTa** version for token classification. We trained the model using the *Trainer*[6] module from huggingface. The training process was conducted over three epochs, using the default cross-entropy loss for classification tasks. The details of the training arguments can be found on the Github repository. The metrics reported during training corresponds to *seqeval* module, similar to the strict mode we are reporting. Furthermore, the report for training process is available on *wandb*[7].

[6]Trainer module link: trainer

[7]MLM training report: report

The results obtained on testing phase using MLM approach can be seen in the table 3. These results correspond to a subset of test data given by the alignment problems with generative approaches. The results over the entire data are available on the repository. We are showing these ones in order to obtain a fair comparison between the different approaches.

Furthermore, the results obtained are inline with those obtained by Tian et al. [26]

3.3.3 Generative approach experiments

The experiments for the Generative approach over the entire dataset were conducted with the same methodology as those explained in Section 3.3.1. In this case, we made predictions over the entire test dataset. Once again, we only considered sentences with the same length after tokenization in true and predicted labels. This resulted in a reduction of around 17%. In this case we kept this data distribution even when it does not match to the one used for the other two approaches, as the performance difference is evident.

The results obtained are shown in Table 3.

3.3.4 MLM + Generative model approach

For this approach, the experiments mainly involve a combination of the two approaches explained before. We take the **RoBERTa** model already trained on the **CHIA** dataset and generate the annotations for the test data. Then, using the same parameters for loading the Mistral-7b model, we perform the verification step with the prompt explained in Section 2.3.

As we have discussed, due to alignment problems between the annotations and the ground truth, we have used a subset of the test data that represents around 70% of its original size for testing.

The results are shown in table 3.

3.3.5 Discussion

In the results reported, we can observe a clear difference between the two approaches using MLM and the pure generative one. It suggests that, in a *Few-shot* scenario, the performance of generative models for NER on biomedical data is limited, which is consistent with the results and analysis made by Naguib et al. [23].

However, we can see that in the third approach, the use of a generative approach to make a *verification* process to the MLM's annotations shows interesting results, mainly for *strict* evaluation and also for overall precision. It strengthens our hypothesis

			Strict			Relaxed		
Entity	Model	acc	recall	prec	f1-score	recall	prec	f1-score
overall	RoBERTa	**0.8273**	0.6960	0.6645	0.6799	**0.8361**	0.7982	0.8167
	Mistral-7b	0.6011	0.5124	0.5346	0.5232	0.6283	0.6556	0.6417
	RoBERTa + Mistral-7b	0.8156	**0.7389**	**0.7353**	**0.7371**	0.8349	**0.8308**	**0.8329**
Person	RoBERTa	-	**0.8613**	**0.8130**	**0.8365**	**0.8613**	**0.8130**	**0.8365**
	Mistral-7b	-	0.0224	0.0285	0.0251	0.0449	0.0571	0.0503
	RoBERTa + Mistral-7b	-	0.7701	0.7444	0.7570	0.8160	0.7888	0.8022
Drug	RoBERTa	-	0.7670	0.7460	0.7564	0.8714	0.8476	0.8594
	Mistral-7b	-	0.5759	0.5301	0.5520	0.6937	0.6385	0.6649
	RoBERTa + Mistral-7b	-	**0.7854**	**0.7548**	**0.7698**	**0.8866**	**0.8521**	**0.8690**
Value	RoBERTa	-	**0.7607**	0.7227	**0.7412**	**0.9043**	**0.8590**	**0.8811**
	Mistral-7b	-	0.0099	0.1363	0.0185	0.0598	0.8181	0.1114
	RoBERTa + Mistral-7b	-	0.7549	**0.7230**	0.7386	0.8627	0.8262	0.8441
Condition	RoBERTa	-	0.7722	0.6894	0.7284	**0.9204**	0.8217	**0.8682**
	Mistral-7b	-	0.6465	0.6060	0.6256	0.7748	0.7263	0.7498
	RoBERTa + Mistral-7b	-	**0.7836**	**0.7823**	**0.7829**	0.8681	**0.8668**	0.8675
Measurement	RoBERTa	-	0.6441	0.6176	0.6306	**0.8343**	**0.8000**	**0.8168**
	Mistral-7b	-	0.1607	0.1223	0.1389	0.3137	0.2388	0.2711
	RoBERTa + Mistral-7b	-	**0.6625**	**0.6315**	**0.6467**	0.8098	0.7719	0.7904
Temporal	RoBERTa	-	**0.6038**	0.5670	0.5849	**0.7727**	**0.7256**	**0.7484**
	Mistral-7b	-	0.0000	0.0000	0.0000	0.0578	0.6086	0.1056
	RoBERTa + Mistral-7b	-	0.5783	**0.6233**	**0.6000**	0.7727	0.7168	0.7437
Procedure	RoBERTa	-	0.5524	0.5302	0.5410	0.7342	**0.7046**	0.7191
	Mistral-7b	-	0.3745	0.3983	0.3860	0.4741	0.5042	0.4887
	RoBERTa + Mistral-7b	-	**0.6142**	**0.5771**	**0.5951**	**0.7500**	0.7046	**0.7266**
Device	RoBERTa	-	**0.3000**	**0.375**	**0.3333**	**0.3000**	**0.3750**	**0.3333**
	Mistral-7b	-	0.1515	0.2777	0.1960	0.1515	0.2777	0.1960
	RoBERTa + Mistral-7b	-	0.3000	0.3333	0.3157	0.3000	0.3333	0.3157
Observation	RoBERTa	-	0.2	0.3225	0.2469	**0.3400**	**0.5483**	**0.4197**
	Mistral-7b	-	0.0240	0.0306	0.0269	0.0880	0.1122	0.0986
	RoBERTa + Mistral-7b	-	**0.2142**	**0.3500**	**0.2658**	0.3265	0.5333	0.4050

Table 3: Results of three approaches over test dataset.

in Section 2.3 about the capacity of generative models to work better on a verification task than in a fully annotated schema.

It would be important to get a more accurate idea of the real impact of this approach to solve the alignment problems faced during the experiments. Testing over the entire dataset would be important, mainly for less common entities.

Furthermore, exploring other approaches for verification prompts would be interesting. Making clearer prompts, with fewer entities at a time, can improve the efficiency of the generative model [23].

It is important to consider the big difference in terms of computational resource demand between MLM and generative models. The second ones are really *expensive*, and even when you can obtain slightly better results, this topic is something to keep in mind. There is a significant difference in the computation time during the inference phase between the two approaches. For the MLM experiments, using a batch size of eight resulted in 164 batches, and the generation took less than a minute. On the other hand, for the generative approach experiments, using a batch size of one, the average time per sentence was around forty seconds.

The experiments were performed using a Google Colab environment with a single T4-GPU. We consider it would be more accurate to conduct the experiments using more GPUs to enable more efficient generation for the second set of experiments.

4 Conclusions

In this work, we explored the capacity of generative models to enhance the performance of current models on NER in the context of biomedical data. We proposed two generative approaches for the task: the first being a pure generative approach, and the second involving a combination of a MLM with a generative model to verify the annotations. The results obtained suggest that this verification step, conducted after MLM annotations, could be a promising way to leverage the potential of each model to improve annotations. However, conducting more rigorous experiments would be important to obtain more definitive results.

References

[1] Yadav V, Bethard S. A survey on recent advances in named entity recognition from deep learning models. arXiv preprint arXiv:1910.11470. 2019 Oct 25.

[2] Grishman R, Sundheim BM. Message understanding conference-6: A brief history. InCOLING 1996 volume 1: The 16th international conference on computational linguistics 1996.

[3] Mollá D, Van Zaanen M, Smith D. Named entity recognition for question answering. InAustralasian Language Technology Association Workshop 2006 (pp. 51-58). Australasian Language Technology Association.

[4] Babych B, Hartley A. Improving machine translation quality with automatic named entity recognition. In Proceedings of the 7th International EAMT workshop on MT and other language technology tools, Improving MT through other language technology tools, Resource and tools for building MT at EACL 2003.

[5] Yadav V, Bethard S. A survey on recent advances in named entity recognition from deep learning models. arXiv preprint arXiv:1910.11470. 2019 Oct 25.

[6] Guo J, Xu G, Cheng X, Li H. Named entity recognition in query. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval 2009 Jul 19 (pp. 267-274).

[7] Weston L, Tshitoyan V, Dagdelen J, Kononova O, Trewartha A, Persson KA, Ceder G, Jain A. Named entity recognition and normalization applied to large-scale information extraction from the materials science literature. Journal of chemical information and modeling. 2019 Jul 31;59(9):3692-702.

[8] Raza S, Schwartz B. Detecting biomedical named entities in COVID-19 texts. InWorkshop on Healthcare AI and COVID-19 2022 Jul 21 (pp. 117-126). PMLR.

[9] Cho H, Lee H. Biomedical named entity recognition using deep neural networks with contextual information. BMC bioinformatics. 2019 Dec;20:1-1.

[10] Kury F, Butler A, Yuan C, Fu LH, Sun Y, Liu H, Sim I, Carini S, Weng C. Chia, a large annotated corpus of clinical trial eligibility criteria. Scientific data. 2020 Aug 27;7(1):281.

[11] Hanisch D, Fundel K, Mevissen HT, Zimmer R, Fluck J. ProMiner: rule-based protein and gene entity recognition. BMC bioinformatics. 2005 May;6:1-9.

[12] Hearst MA, Dumais ST, Osuna E, Platt J, Scholkopf B. Support vector machines. IEEE Intelligent Systems and their applications. 1998 Jul;13(4):18-28.

[13] Vlachos A. Evaluating and combining and biomedical named entity recognition systems. InBiological, translational, and clinical language processing 2007 Jun (pp. 199-200).

[14] Lafferty J, McCallum A, Pereira F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Icml 2001 Jun 28 (Vol. 1, No. 2, p. 3).

[15] Singh TD, Nongmeikapam K, Ekbal A, Bandyopadhyay S. Named entity recognition for manipuri using support vector machine. InProceedings of the 23rd Pacific Asia Conference on Language, Information and Computation, Volume 2 2009 Dec (pp. 811-818).

[16] Morwal S, Jahan N, Chopra D. Named entity recognition using hidden Markov model (HMM). International Journal on Natural Language Computing (IJNLC) Vol. 2012;1.

[17] Das A, Garain U. Crf-based named entity recognition@ icon 2013. arXiv preprint arXiv:1409.8008. 2014 Sep 29.

[18] Sherstinsky A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena. 2020 Mar 1;404:132306.

[19] Chung J, Gulcehre C, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555. 2014 Dec 11.

[20] Luo L, Yang Z, Yang P, Zhang Y, Wang L, Lin H, Wang J. An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition. Bioinformatics. 2018 Apr 15;34(8):1381-8.

[21] Ashish V. Attention is all you need. Advances in neural information processing systems. 2017;30:I.

[22] Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. 2018 Oct 11.

[23] Naguib M, Tannier X, Névéol A. Few shot clinical entity recognition in three languages: Masked language models outperform LLM prompting. arXiv preprint arXiv:2402.12801. 2024 Feb 20.

[24] Ashok D, Lipton ZC. Promptner: Prompting for named entity recognition. arXiv preprint arXiv:2305.15444. 2023 May 24.

[25] Hu Y, Chen Q, Du J, Peng X, Keloth VK, Zuo X, Zhou Y, Li Z, Jiang X, Lu Z, Roberts K. Improving large language models for clinical named entity recognition via prompt engineering. Journal of the American Medical Informatics Association. 2024 Jan 27:ocad259.

[26] Tian S, Erdengasileng A, Yang X, Guo Y, Wu Y, Zhang J, Bian J, He Z. Transformer-based named entity recognition for parsing clinical trial eligibility criteria. InProceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics 2021 Aug 1 (pp. 1-6).

[27] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692. 2019 Jul 26.

[28] Jiang AQ, Sablayrolles A, Mensch A, Bamford C, Chaplot DS, Casas DD, Bressand F, Lengyel G, Lample G, Saulnier L, Lavaud LR. Mistral 7B. arXiv preprint arXiv:2310.06825. 2023 Oct 10.

[29] Dettmers T, Pagnoni A, Holtzman A, Zettlemoyer L. Qlora: Efficient finetuning of quantized LLMs. Advances in Neural Information Processing Systems. 2024 Feb 13;36.

A Appendix: Prompts used in generative approaches

```
First prompt:
I need to perform a named entity recognition task on a  text related with inclusion criteria in clinical trials.

The entities you need to recognize are: Condition, Value, Drug, Procedure, Measurement, Temporal, Observation,
Person, Mood, Device and Pregnancy_considerations.

Particularly you have to produce the ouput in the BIO format. I will show you an example of the expected output.

Input text: Patients with symptomatic CNS metastases or leptomeningeal involvement

Output:
    Patients O
    with O
    symptomatic O
    CNS B-Condition
    metastases I-Condition
    or O
    leptomeningeal B-Condition
    involvement I-Condition

You can see that tokens without any entity are labeled as O, and the tokens that are part of an entity are
labeled as B-<entity> or I-<entity> depending on if they are the beginning or the inside of the entity.

Please, just answer the question for this specific example and stop writting after that.

Input text: 18 years or older patients who are proven to be infected by Helicobacter pylori based on positive
in Urea Breath Test or positive in histopathologic examination of biopsy in antrum and corpus of gaster through
esophagoduodenoscopy .
```

Figure 1: Example of the first prompt proposed for Generative approach.

```
Second prompt:
I am working on a named entity recognition problem, in the context of clinical
trials eligibility criteria. I will show you the list of entities:

- Condition, Value, Drug, Procedure, Measurement, Temporal, Observation, Person, Device

Your task consists in annotate the named entities in a given sentence in the format I will explain you.

I will explain you with some examples:

Example 1:
Input: Patients who have received prior chemotherapy for unresectable disease.
Output: Patients who have received prior <Procedure>chemotherapy</Procedure> for <Condition>unresectable
disease</Condition>.

Example 2:

Input: Patients with any other severe concurrent disease, which in the judgment of the investigator, would make
the patient inappropriate for entry into this study.
Ouput: Patients with any other severe <Condition>concurrent disease</Condition>, which in the judgment of the
investigator, would make the patient inappropriate for <Observation>entry into this study</Observation>.

As you can see, in each example, the extracted entities are enclosed using the sintax: <ENT>text of the entity</
ENT>.
Please now annotate as explained before the following sentence:
Input: 18 years or older patients who are proven to be infected by Helicobacter pylori based on positive in
Urea Breath Test or positive in histopathologic examination of biopsy in antrum and corpus of gaster through
esophagoduodenoscopy .
Output:
```

Figure 2: Example of the second prompt proposed for Generative approach.

```
I am working in a named entity recognition task for Clinical trial
eligibility criteria. I have some annotations I need you to check. The list of possible entities is
Condition,Person,Device,Procedure,Value,Drug,Temporal,Observation,Measurement.
I want you to check the correcteness of the sentence and return the correct annotations in the exact same
format as the original annotations.
Please use just the entities in the list. I will show you a first simple example for you to understand the
task:
input: The patient has a <Condition>fever</Condition> of 38 degrees Celsius .
output: The patient has a <Condition>fever</Condition> of <Value>38 degrees Celsius</Value> .

Now it is your turn. Please check the following sentence:
input: The patient has a <Condition>fever</Condition> of 38 degrees Celsius .
output:
```

Figure 3: Example of the verification prompt used in MLM + Generative model approach.