Co-Evolutionary Prompt Optimization for Improving Language Model Performance on Specialized Domains

Abstract

Prompt engineering is a popular customization method for language models, particularly relevant in the case of tasks and domains with limited access to annotated data for model fine-tuning. Still, the discovery of effective prompts is challenging, driving a desire for general prompt learning methods. This paper advances *CoEvo*, i.e. a prompt learning approach that combines ideas from co-evolutionary computation together with the use of relatively small language models for data selection, and for emulating genetic crossover and mutation. We evaluate *CoEvo* on four tasks involving clinical or legal text, comparing different prompting techniques. The results show that *CoEvo* is capable of discovering effective and human-understandable prompts, with improvements over initial prompts designed manually. The code for replicating our experiments will be made available, upon acceptance.

1 Introduction

Large Language Models (LLMs) have shown impressive abilities in various domains and tasks, although their use often depends on the design of adequate instruction prompts. LLMs can be seen as black-box computers that execute programs specified by natural language instructions (Zhou et al., 2022). While LLMs can address a broad range of tasks, the way the programs are processed is not intuitive for humans, and the quality of a prompt can only be measured after its execution. Prompt engineering, which entails the crafting of task-specific instructions in natural language, has thus become a central concern associated with the use of LLMs. This is particularly true for specialized tasks and domains, with limited access to the annotated data that can support model fine-tuning.

Recent studies have focused on understanding the semantic and contextual nuances in prompts, examining how subtle changes can lead to significantly different responses from LLMs, and in some cases presenting comprehensive sets of principles that can improve prompt quality (Bsharat et al., 2023; White et al., 2023). However, the task of prompting LLMs to produce specific responses, making full use of their capabilities, continues to pose a considerable challenge. These difficulties have driven a desire for general prompt learning methods, corresponding to automatic optimization strategies that (i) are gradient-free and capable of working with discrete prompts composed of sequences of word tokens, (ii) can work with black-box LLMs, (iii) are computationally efficient, and (iv) are interpretable, all at once.

Considering the aforementioned properties, this paper advances *CoEvo* as a meta-heuristic approach for prompt learning, combining co-evolutionary computation with the use of relatively small language models for data selection, and for implementing text transformations that emulate the genetic operators of crossover and mutation. In our approach, the task prompts are broken down into multiple components, which are independently evolved through crossover/mutation, later being concatenated to compose the complete task prompt. By breaking down large prompts into smaller and simpler components, we argue that the proposed approach can better explore the capabilities of smaller language models, instead of assuming the use of highly-capable LLMs such as GPT-4 (OpenAI, Achiam J, et al., 2024). Being both simple and generalist, the proposed approach can significantly reduce the human effort required to create and validate effective prompts.

We evaluate *CoEvo* using different prompting strategies on four tasks from two specialized domains (two natural language inference tasks in the clinical and legal domains, plus two specialized summarization tasks, also involving clinical and legal documents), using the

`Llama-3.2-3B-Instruct`[1] open-source language model. To reduce computational costs, while simulating a context with data constraints, we only use 25% of the available data in our experiments. Additionally, we compare having the data selected at random, against a selection method based on a data quality score, computed via prompting.

The experimental results show that *CoEvo* is indeed capable of discovering effective and human-understandable prompts. We improve over the results of manually designed prompts, achieving strong task-specific results, without resorting to model fine-tuning.

The rest of this document is organized as follows: Section 2 discusses related work, focusing on previous approaches inspired on evolutionary computation for prompt optimization. Section 3 details *CoEvo*, analyzing different choices associated with its implementation. Section 4 shows the evaluation of *CoEvo* on a set of specialized tasks. Finally, Section 5 summarizes the main conclusions and presents directions for future work.

2 Related Work

A number of recent studies have focused on automating prompt engineering tasks, often relying on optimization meta-heuristics that operate on text through gradient-free approaches.

An example of a relevant previous method is the Automatic Prompt Engineer (APE) (Zhou et al., 2022), which uses iterative and directionless Monte Carlo search over the space of prompts. APE starts by generating prompt candidates, providing to the LLM input/output pairs, and asking the model to infer prompts for the task. The candidates are then evaluated, keeping the top-performing ones. An LLM is also used to paraphrase the best prompts. The selection plus paraphrasing process is repeated until a predetermined stopping criterion is met. We took inspiration from this particular approach to develop a Monte Carlo baseline for our experiments.

Similarly, Gradient-free Instructional Prompt Search (GrIPS) (Prasad et al., 2022) applies greedy search methods to search for better prompts, in this case relying on simple edit operations such as concatenation, swapping, paraphrasing, or deletion. Starting from a set of base instructions that are broken down into clauses, GrIPS generates new instructions through the aforementioned edit operations. The top-k best instructions are kept for the next iteration, and the process repeats until a predetermined stopping criterion is met. GrIPS also explores a simulated annealing approach by sampling a new candidate for the next iteration when there is no improvement from the previous one. An acceptance function is employed to determine if the candidate is accepted. This function is more exploratory in the early iterations and more exploitative in the later ones.

Other studies have instead proposed approaches inspired by evolutionary computation. For example, Genetic Prompt Search (GPS) (Xu et al., 2022) starts with a set of handcrafted prompts. At each iteration, the best prompts are selected. These are used to generate new prompts, with strategies such as back-translation or sentence continuation.

Optimization by PROmpting (OPRO) evolves task-specific prompts using a meta-prompt that instructs a LLM to produce variations (Yang et al., 2023). The meta prompt includes a description of the optimization task, examples from the training data, and an optimization trajectory that records previously evaluated prompts along with their evaluation scores. At each iteration, several new candidates are generated and evaluated. The meta-prompt is also updated with the new results.

EvoPrompt (Guo et al., 2023) corresponds to a prompt learning approach inspired by Evolutionary Algorithms (EAs) that starts with a handcrafted population of task-specific prompts, and uses LLMs to implement the genetic operations. The authors have specifically explored two different strategies. In the first, new task prompts are generated with a crossover prompt applied to two individuals selected based on a roulette wheel method, according to their fitness value. The resulting prompt is then mutated with a mutation prompt. The second approach generates new instruction prompts using a prompt with a four-step plan and three randomly selected task prompts from the population. The goal of each step is as follows: (1) identify the differences between the first and second prompts; (2) mutate the differences; (3) combine the mutated differences with the third prompt; (4) combine the result of the previous step with a very simple predefined prompt for the task. EvoPrompt significantly outperformed both manually designed prompts and APE. On a diverse set of language understanding tasks, using the Alpaca-7B language model, EvoPrompt improved accuracy over manual instructions by 6.1%, whereas

[1] https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct

APE obtained an improvement of 2.7%. On two text summarization tasks, EvoPrompt improved ROUGE-1 by 3.9%, while APE did not improve over manual instructions.

Prompt Optimization with Textual Gradients (ProTeGi) incrementally instructs an LLM to produce text feedback on how to update a previous instruction (Pryzant et al., 2023), drawing inspiration from gradient descent to address the challenge of prompt optimization. This approach begins with a base task prompt (p_0), which is evaluated on a batch of data, while keeping track of the mistakes that are made. The mistakes are then used to prompt the LLM to generate n summaries of what is wrong with the base prompt, considering the errors. Each of the summaries, together with p_0, is then used to generate new prompts. The best performing ones are kept for the next iteration.

PromptBreeder (PB) is also inspired by EAs, taking a meta-learning approach that evolves multiple LLM-generated mutation prompts together with the task prompts (Fernando et al., 2023). Mutated prompts are generated through direct, estimation of distribution, and Lamarckian mutations, as well as crossovers and hypermutations (i.e., mutating one of the existing direct mutation prompts, and then applying this altered version directly to the task prompt). A binary tournament genetic algorithm performs the population updates, in which two individuals are selected and the less effective one is replaced by a mutated version of the more effective one. The specific mutation that is applied is randomly selected. The initial population of task prompts is generated by an LLM. Through experiments using the Palm 2-L model (Rohan Anil, Andrew M. Dai, et al., 2023), the authors show that PB outperforms APE and OPRO. For instance, on the GSM8K benchmark (Cobbe et al., 2021), APE improved over a single chain-of-thought prompt by 21.5%, OPRO improved by 23.8%, and PB improved results by 27.5%.

CoEvo builds upon the aforementioned prior work in prompt optimization methods using EAs, introducing concepts related to cooperative evolution, as a way to better leverage smaller language models. Also, while most previous work focused evaluation on standard benchmark tasks, we target complex tasks in the legal and clinical domains.

```
1 - Task Description
2 - Description for Premise
PRIMARY CLINICAL TRIAL REPORT
(SECONDARY CLINICAL TRIAL REPORT)
3 - Description for Hypothesis Statement
STATEMENT
4 - Answer Description
```

Figure 1: Prompt structure used in a 0-shot prompting strategy for the NLI4CT dataset. The numbered parts correspond to the 4 components that are used. Elements shown in a monospaced font are sample-dependent.

3 Co-Evolutionary Prompt Learning

We propose *CoEvo*, i.e. a co-evolutionary prompt learning algorithm for the automatic optimization of task-specific prompts. In *CoEvo*, we break down large task-specific prompts into multiple components, each detailing a given part of the overall task. The components evolve in isolation, but come together to form individuals, which are then evaluated with task-specific fitness functions. New components are generated from pairs of components by applying crossover and mutation operations in sequence. These operations are done by prompting a relatively small language model, with a mutation or a crossover prompt.

In more detail, for each task, we can define a prompt consisting of several components. For instance, in a Natural Language Inference (NLI) task involving the comparison of a premise and an hypothesis, we can structure the prompt into four components: a general task description, specific descriptions of the premise and the hypothesis, and a description of the expected answer. We initialize *CoEvo* with a hand-crafted set of options for each component, and define the task's fitness function (e.g., the accuracy or the F1-score in NLI tasks, or the ROUGE-1 F1-score in summarization tasks).

Classic evolutionary algorithms use hyper-parameters to balance between exploration and exploitation. This dichotomy also exists in *CoEvo* through hyper-parameters such as the crossover or mutation probabilities. The process for fine-tuning these hyper-parameters is described in Appendix F.

3.1 Proposed Approach

We divide *CoEvo* into two algorithms, where Algorithm 1 deals with creating a population from lists of components, and Algorithm 2 formalizes the co-evolutionary process, leveraging the population from Algorithm 1.

Algorithm 1 Create and evaluate a population $\mathcal{P}$

Require: Number of prompt components C;
Require: Fitness function $f_{\mathcal{D}}(\cdot)$, that uses a dataset $\mathcal{D}$ for evaluating a population of individual prompts $p = [s_1, \ldots, s_C]$;
Require: List $\mathbf{A} = [\mathcal{A}_1, \ldots, \mathcal{A}_C]$ of alternatives $\mathcal{A}_z$ for each prompt component $1 \leq z \leq C$;
Require: Population size N.

$\mathcal{P} = \{\}$
$\epsilon =$ The empty string
for $i \in \{1, \ldots, N\}$ **do**
  $p = [s_1 = \epsilon, \ldots, s_C = \epsilon]$
  for $j \in \{1, \ldots, C\}$ **do**
    $p[j] \sim_{\text{Uniform}} \mathbf{A}[j]$
  end for
  $\mathcal{P} = \mathcal{P} \cup \{\langle p, f_{\mathcal{D}}(p)\rangle\}$
end for
return $\mathcal{P}$

Algorithm 1 constructs a population for a specific task and prompting strategy, with a predefined number of components C. Figure 1 outlines the structure for one prompting strategy that we used with a clinical NLI dataset (Jullien et al., 2023). For this particular setting and dataset, we have 4 components (i.e., task description, premise description, hypothesis statement description, and answer description). The algorithm takes a list of lists of strings $\mathbf{A}$ as input. Each element $\mathcal{A}_z \in \mathbf{A}$ contains the list of possible strings that can be used to describe the corresponding component. The population will then consist of N individuals, each with components randomly selected from the list of possibilities $\mathcal{A}_z$. Repetitions are allowed, meaning that the same description can appear in multiple individuals. The individuals are evaluated using a fitness function $f_{\mathcal{D}}(\cdot)$. At the end of this process, each individual has a single description for each component, and a score derived from the evaluation of all its components combined.

Algorithm 2 outlines the evolution of a population through generational replacement with elitism, starting with an initial population (i.e., the output from Algorithm 1). An iteration of the algorithm involves: saving the top-E performing individuals from the current population, where E stands for the elite population size; creating new lists of strings for each component; creating a new population from the newly generated lists; and combining these new individuals with the elite population. For the j-th component, each new description is generated by: sampling two individuals (based on the evaluation scores) from the population; taking their j-th component; performing a crossover operation with probability p_c; and performing a mutation operation with probability p_m.

The hyper-parameters of the algorithm include: the population size N; the elite population size E; the crossover operator $f_C(\cdot, \cdot)$; the mutation operator $f_M(\cdot)$; the crossover probability p_c; the mutation probability p_m; and the sampling temperature S_T. Considering $E > 0$ allows us to guarantee that a number of top-performing individuals are always kept for the next iteration. Mutation and crossover operations are performed by LLMs. For each operation, we considered 5 different prompts to describe the corresponding operation, selected at random every time they were used. Appendix B showcases the operator prompts.

We use a roulette wheel selection strategy, with sampling temperature S_T. We sample the individuals, for the combinaion of their components, by dividing their fitness score by S_T and then applying a `softmax` transformation. A lower S_T value results in a more exploitative strategy, whereas a higher value encourages exploration. Higher probabilities for the mutation and crossover operations can also be used to promote exploration.

3.2 Monte-Carlo Baseline

To better assess the performance of *CoEvo* and to compare it against a strategy similar to APE (Zhou et al., 2022), we developed a Monte-Carlo (`MC`) baseline. In this baseline, we perform a non-informed prompt search by setting $E = 0$, $p_c = 0$, $p_m = 0.5$, and sampling individuals from the population at random (instead of sampling individuals based on their fitness scores).

4 Experiments

We now describe the experimental evaluation of the proposed approach.

4.1 Evaluation Methodology

We assess the performance of *CoEvo* on four benchmark tasks in the clinical and legal domains. A brief description of the datasets and prompting strategies that were used is given next.

- **NLI4CT** (Jullien et al., 2023): A natural language inference task for clinical trial data. Instances contain one or two Clinical Trial Reports (CTRs) as the premise, and a clinical

Algorithm 2 Find the best individual prompt by evolving a population

Require: All requirements from **Algorithm 1**;
Require: Maximum number of iterations T;
Require: Number of patience iterations W;
Require: Elite population size E;
Require: Mutation operator $f_M(\cdot)$;
Require: Mutation probability p_m;
Require: Crossover operator $f_C(\cdot, \cdot)$;
Require: Crossover probability p_c;
Require: Sampling temperature S_T;

$\mathcal{P} =$ **Algorithm 1**$(C, f_{\mathcal{D}}(\cdot), \mathbf{A}, N)$
$t = 1$
$w = 1$
$f_{\mathcal{D}}^{\max} = 0$
while $(t \leq T) \wedge (w \leq W)$ **do**
  $\mathcal{P} \leftarrow \text{sort}(\mathcal{P}, \geq f_{\mathcal{D}}(\cdot))$
  $\mathcal{P}^{Elite} \leftarrow \mathcal{P}[:E]$
  $\mathbf{A}' = [\mathcal{A}'_1 = \emptyset, \ldots, \mathcal{A}'_C = \emptyset]$
  for $j \in \{1, \ldots, C\}$ **do**
    for $i \in \{1, \ldots, len(A_j)\}$ **do**
      $\text{Dist} = \{f_{\mathcal{D}} : \langle \cdot, f_D \rangle \in \mathcal{P}\}$
      $\text{Dist} = \text{softmax}(\text{Dist}/S_T)$
      $\langle p_1, f_{\mathcal{D}}(p_1) \rangle \sim_{\text{Dist}} \mathcal{P}$
      $s = p_1[j]$
      if $\text{random}() \leq p_c$ **then**
        $\langle p_2, f_{\mathcal{D}}(p_2) \rangle \sim_{\text{Dist}} (\mathcal{P} - \{\langle p_1, f_{\mathcal{D}}(p_1) \rangle\})$
        $s = f_C(s, p_2[j])$
      end if
      if $\text{random}() \leq p_m$ **then**
        $s = f_M(s)$
      end if
      $\mathbf{A}'[j] = \mathbf{A}'[j] \cup \{s\}$
    end for
  end for
  $\mathcal{P} = \mathcal{P}^{Elite} \cup$ **Alg. 1**$(C, f_{\mathcal{D}}(\cdot), \mathbf{A}', N - E)$
  if $\max(\{f_{\mathcal{D}} : \langle \cdot, f_{\mathcal{D}} \rangle \in \mathcal{P}\}) > f_{\mathcal{D}}^{\max}$ **then**
    $f_{\mathcal{D}}^{\max} = \max(\{f_{\mathcal{D}} : \langle \cdot, f_{\mathcal{D}}(\cdot) \rangle \in \mathcal{P}\})$
    $w = 1$
  else
    $w = w + 1$
  end if
  $t = t + 1$
end while
$\mathcal{P}^* = \{\langle p^*, f_{\mathcal{D}} \rangle \in \mathcal{P} : f_{\mathcal{D}} = f_{\mathcal{D}}^{\max}\}$
return p^*

statement as the hypothesis. The possible labels for each instance are either "Entailment" or "Contradiction". For $f_{\mathcal{D}}(\cdot)$, we use the macro F1-score. Exclusively for the test set, NLI4CT uses two additional metrics, namely faithfulness and consistency. These metrics were proposed to assess how well a system deals with semantic-altering and semantic-preserving interventions to the hypothesis, respectively. In this task, we explore two prompting strategies: (i) 0-shot, using 4 components, as shown in Figure 1; (ii) HR, with an additional component highlighting parts of the CTRs retrieved via embeddings for each statement, together with a two-step self-reasoning approach, resulting in 7 components in total. We detail these prompting strategies and the retrieval method in Appendix A.1.

- **ContractNLI** (Koreeda and Manning, 2021): A document-level natural language inference task for Non-Disclosure Agreements (NDAs), which are divided into textual spans. Instances have an NDA as the premise, a statement as the hypothesis, and a list of oracle spans that support the hypothesis. We explore the version of the task where the possible labels for each instance are either "Entailment" or "Contradiction". For $f_{\mathcal{D}}(\cdot)$ we use accuracy, although we also report the F1-scores for the test set. The F1-scores are first calculated at a document level and then averaged out across documents, as reported by Koreeda and Manning (2021). In this task, we explore a single prompting strategy, highlighting either the oracle spans or spans that were retrieved via embeddings for each statement. Additional details are given in Appendix A.2.
- **MEDIQA-CHAT** (Yim et al., 2023): A full dialogue summarization task. Instances include a doctor-patient dialogue and a target summary, consisting of a single clinical note. For $f_{\mathcal{D}}(\cdot)$, we use the ROUGE-1 F1-score, while also reporting the ROUGE-2, ROUGE-L, and ROUGE-LSum F1-scores for the test set. In this task, we use a 1-shot prompting strategy based on the approach proposed by Giorgi et al. (2023), using 4 components. For each instance, we retrieve the most similar clinical note from the available data via embeddings applied to the dialogues. Additional details are given in Appendix A.4.
- **ToS-Sum** (Manor and Li, 2019): A plain english summarization task focusing on contracts. Instances include a legal text snippet

from a Terms of Conditions (ToS) contract, and a target summary. For $f_{\mathcal{D}}(\cdot)$, we use the ROUGE-1 F1-score, while also reporting the ROUGE-2 and ROUGE-L F1-scores for the test set. In this task, we use a 1-shot prompting strategy similar to the one used in the MEDIQA-CHAT task. Additional details are given in Appendix A.3.

In the MEDIQA-CHAT and ToS-Sum tasks, we also allow for the use of a 0-shot prompting strategy, by including an empty string element in the initial set of descriptions for the component responsible for the 1-shot strategy. When the empty string is selected, the component is omitted, and a 0-shot prompting strategy is used. Similarly, we also include the empty string in the components of the HR prompting strategy for the NLI4CT task. This approach allows *CoEvo* to revert to a simpler prompting strategy if this is deemed optimal.

4.2 Experimental Setup

In all the *CoEvo* experiments, we use the following hyper-parameter values: $N = 25$, $E = 5$, $p_M = p_C = 0,25$, and $S_T = 10,0$. These were obtained after an initial set of tests involving a grid-search analysis, as detailed in Appendix F. For the mutation and crossover operators, every time an operation is performed, we randomly select one among 5 prompts defined *a priori*. We manually designed each of these prompts with a different goal in mind (e.g., by asking the LLM to only make small edits, or instead saying it can make as many edits as it finds appropriate). We detail these prompts in Appendix B.

In every experiment, we start with 5 handcrafted options from each of the prompt components. To increase variability, we use different tones and phrasings, and provide more or less details in each alternative. We provide examples in Appendix A.

NLP tasks in the clinical and legal domains typically face data availability challenges. As a way to further mimic these conditions and reduce computational costs in our experiments, we only used 25% of the available data for each task. We explore a data selection process based on a data quality score and compare it with a random selection method. We assess the data quality score of every instance using a prompt-based approach inspired by Ask-LLM (Sachdeva et al., 2024). The approach used to assess the data quality is detailed in Appendix C.

We use the `Llama-3.2-3B-Instruct`[1] language model for all *CoEvo* inferences. For the NLI tasks, we classify each instance according to the probability of generating the *Yes* or *No* tokens. For the summarization tasks, we employ a beam search decoding algorithm with width 3, as this offers an effective balance between computational efficiency and performance (Cohen and Beck, 2019).

4.3 Experimental Results

In this section, we report on the experiments conducted to assess the performance of *CoEvo* in the datasets described in Section 4.1. To this end, we compare the prompts optimized using *CoEvo* with state-of-the-art approaches for each task, as well as the prompts optimized with the `MC` baseline. The prompts resulting from *CoEvo* are also evaluated with a larger LLM, specifically `Llama3.3-70B-Instruct`[2] using 4-bit quantization, in an attempt to assess if computationally efficient optimization can inform the prompting of larger models (e.g., Hui et al. (2024) had already shown benefits in doing prompt optimization with small models). The reported results follow the evaluation methodologies of prior work, and the evaluation metrics are reported for the optimized prompt, obtained after each experiment, on the test sets of the corresponding tasks.

Table 1 summarizes the results on the NLI4CT dataset, comparing the best systems, in each metric, over the 2024 edition of the NLI4CT shared task (trained on all the available data), against six versions of *CoEvo* and two of `MC`. When using *CoEvo* with data filtered by data quality, we obtain greater improvements compared to using randomly selected data (5% vs 3% of improvement in accuracy, over the average individual in the initial population). *CoEvo* also outperforms the `MC` baseline, although the results are still far from the best systems at the shared task. The simpler prompting strategy clearly outperformed HR, which in all settings ended up producing worse results.

Figure 2 shows two evolutionary runs for the NLI4CT *CoEvo*$_{\text{(0-shot)}}$ experiments, using 25% of the available data selected at random (at the top) or by data quality (at the bottom). Using the data filtered by data quality leads to a lower initial F1-score, but *CoEvo* is still able to successfully guide the search. Furthermore, model performance using the quality filtered data generalizes better to the test set, as shown in Table 1. The results for the

[2] `https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct`

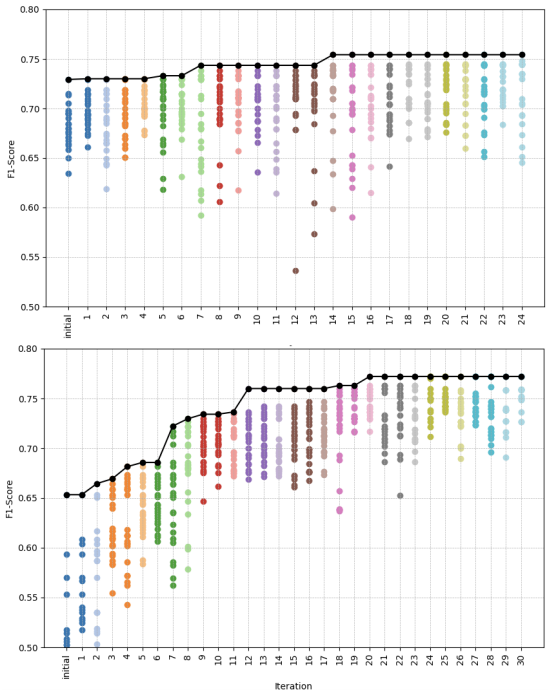

Figure 2: Two evolutionary runs for the NLI4CT CoEvo (0-shot) experiment, using 25% of the available data selected at random (at the top), or using 25% of the available data filtered by the data quality score (at the bottom). Each dot corresponds to an individual. Results for these experiments are also reported on Table 1.

Data	Approach	**F1**	Faith.	Consis.
All	Best Consis.	0.78	0.92	**0.81**
	Best Faith.	0.78	**0.95**	0.78
	Best F1	**0.80**	0.90	0.73
-	Best initial (0-shot)	0.63	0.23	0.83
	Avg. Initial (0-shot)	0.60	0.54	0.76
	Best Initial (0-shot) (70B)	0.76	**0.92**	0.83
25% Ran.	MC (0-shot)	0.63	0.64	0.71
	CoEvo (0-shot)	0.63	0.64	0.62
	CoEvo (0-shot) (70B)	**0.77**	0.91	0.83
	CoEvo (HR)	0.52	0.80	0.76
	CoEvo (HR) (70B)	0.74	0.89	0.72
25% DQ	MC (0-shot)	0.54	0.77	0.76
	CoEvo (0-shot)	0.65	0.60	0.72
	CoEvo (0-shot) (70B)	0.73	0.91	**0.85**
	CoEvo (HR)	0.59	0.67	0.69
	CoEvo (HR) (70B)	0.76	0.91	0.84

Table 1: Results on the NLI4CT task.

remaining prompting strategies and datasets follow similar trends and are shown in Appendix D.

Table 2 summarizes results on the ContractNLI dataset, comparing the approaches reported by Koreeda and Manning (2021), which use all the available data, against four versions of CoEvo and two versions of MC. We obtained competitive results compared to the prior work that fine-tuned models on all the available data. All CoEvo (Oracle) experiments showed improvements over the initial

Data	Approach	**Acc.**	F1 (C)	F1 (E)
All	Majority Vote	0.814	0.239	0.645
	Span NLI (BERTbase)	0.883	0.490	0.795
	Span NLI (BERTlarge)	0.899	0.492	**0.820**
	Oracle NLI (BERTbase)	**0.918**	**0.657**	0.816
	Oracle NLI (BERTlarge)	0.908	0.620	0.806
-	Best Initial (Oracle)	0.847	0.491	0.907
	Avg. Initial (Oracle)	0.667	0.459	0.737
	Best Initial (Oracle) (70B)	**0.931**	0.743	0.950
25% Ran.	MC (Oracle)	0.852	0.556	0.906
	CoEvo (Oracle)	0.877	0.502	0.920
	CoEvo (Oracle) (70B)	**0.931**	**0.744**	**0.955**
	CoEvo (Retrieved)	0.858	0.341	0.911
	CoEvo (Retrieved) (70B)	0.878	0.574	0.922
25% DQ	MC (Oracle)	0.806	0.567	0.869
	CoEvo (Oracle)	0.883	0.481	0.925
	CoEvo (Oracle) (70B)	0.902	0.717	0.920
	CoEvo (Retrieved)	0.830	0.320	0.897
	CoEvo (Retrieved) (70B)	0.879	0.530	0.924

Table 2: Results on the ContractNLI task.

Data	Approach	**R1**	R2	RL
-	TestRank	23.88	6.96	16.96
	KLSum	23.25	6.76	16.67
	Lead-1	24.05	7.30	17.22
	Lead-K	**24.47**	**7.40**	**17.66**
	Random-K	22.39	6.17	16.01
-	Best Initial	31.68	13.47	27.20
	Avg. Initial	23.97	8.01	19.39
	Best Initial (70B)	31.80	13.24	27.08
25% Ran.	MC	30.65	12.44	25.84
	CoEvo	32.25	14.21	28.01
	CoEvo (70B)	30.13	11.49	25.41
25% DQ	MC	31.87	13.59	27.22
	CoEvo	33.68	**16.62**	**29.73**
	CoEvo (70B)	**34.23**	16.13	29.39

Table 3: Results on the ToS-Sum task.

population, with the most notable gains observed when using data filtered by data quality. Specifically, the CoEvo (Oracle) experiment, utilizing the quality-filtered data, led to a 21.6% improvement in accuracy over the average individual in the initial population, and a 3.6% improvement over the best individual in the initial population.

Table 3 summarizes results on the ToS-Sum task, comparing the unsupervised extraction methods reported by Manor and Li (2019) against two versions of CoEvo and two versions of MC. The MC and CoEvo runs outperform all uninformed extractive methods. When using data quality filtering (which outperformed random selection), CoEvo achieved a 9.71% improvement in the ROUGE-1 F1-score over the average individual in the initial population. CoEvo also outperformed the MC baseline, as well as the best initial prompt.

Finally, Table 4 summarizes results on the MEDIQA-CHAT task (Abacha et al., 2023), com-

Data	Approach	**R1**	R2	RL	RLSum
-	Best System	**61.41**	**32.88**	**38.15**	**55.15**
	ChatGPT	47.44	19.01	27.11	39.02
	GPT-4	51.76	22.58	30.29	42.56
-	Best Ini.	49.56	23.41	30.84	46.30
	Avg. Ini.	45.06	19.71	26.93	41.94
	Best Ini. (70B)	56.17	**24.72**	32.85	50.06
25% Ran.	MC	49.88	23.40	30.74	46.54
	CoEvo	50.55	23.33	31.74	47.19
	CoEvo (70B)	**56.33**	24.59	**33.14**	**50.40**
25% DQ	MC	51.33	24.12	31.96	47.71
	CoEvo	50.08	23.48	31.22	46.54
	CoEvo (70B)	55.49	24.18	32.19	49.55

Table 4: Results on the MEDIQA-CHAT task.

paring the best systems that participated in the 2023 shared task, which used all the available data, with *CoEvo* and MC. We obtained our best results with *CoEvo* (1-shot) using data selected at random.

In Appendix E, we show the optimized prompts obtained using *CoEvo*. For each dataset and prompting strategy, we show the prompt that obtained the best results in the test set. We conducted a manual analysis in an attempt to perceive any patterns regarding the optimized prompts that are obtained at the end of the *CoEvo* process. Compared to the initial prompts, the optimized versions can become either simpler in terms of phrasing or more detailed in terms of descriptions. There are cases where the optimized prompt features repetitions, and we also noted that simpler prompting strategies were often preferred (e.g., ignoring the components that used retrieved information).

In some instances, *CoEvo* did not improve on the best individual from the initial population. After analyzing the evolutionary plots of the runs (see Appendix D), we hypothesize that *CoEvo* is prone to overfitting, considering the decrease in performance from the data that guides the optimization, compared to the test set. Employing the data quality filter, this overfitting problem was mitigated. This validates the effectiveness a data quality selection method can have, as a way to create a better sample to assess the performance of the models.

We also note that, in all tasks, *CoEvo* consistently generated prompts that outperform the average individual in the initial population. When using randomly selected data, *CoEvo* achieved the following performance gains over the average individual in the initial population: 3% on NLI4CT, 21% on ContractNLI, 8.28% on ToS-Sum, and 5.49% on MEDIQA-CHAT. Similarly, with the data selected by data quality, the improvements were 5% on NLI4CT, 21.6% on ContractNLI, 9.71% on ToS-Sum, and 5.02% on MEDIQA-CHAT. Although previous work on prompt optimization used different language models and datasets, we can say that the improvements we obtained align with the results reported for APE, EvoPrompt, PB, and OPRO, which all used larger language models. In these studies, the optimized prompts are compared to manually crafted prompts, with improvements ranging from 2.7% to 27.5%, depending on the task as discussed in Section 2.

Lastly, the improvements observed on the smaller model with *CoEvo* did not consistently translate to the larger 70B-parameter model. This discrepancy suggests that the performance gains achieved with smaller models may not always scale proportionally with larger models, which often achieve good results with prompts that are much less effective with small models.

5 Conclusions and Future Work

This paper proposed and evaluated *CoEvo*, i.e. a meta-heuristic approach for prompt learning, combining ideas from co-evolutionary computation together with the use of language models for realizing text transformations that emulate the genetic operators of crossover and mutation. Experiments on four different tasks, from two specialized domains (i.e., clinical and legal), show that the proposed approach is capable of discovering prompts that are effective and human-understandable, leveraging relatively small language models for prompt optimization. Additionally, our experiments show that a data selection method based on prompting can be an effective way of finding the most representative instances of a dataset, in order to efficiently and effectively guide prompt optimization.

Despite the interesting results, there are also many possibilities for further improvement. These include a better analysis of the balance between exploitation and exploration, and the study of techniques to incorporate richer feedback about errors (i.e., features that distinguish between high-quality and low-quality prompts generated during the optimization trajectory), going beyond aggregated evaluation metrics such as accuracy over a training set. Another possible direction for improvement concerns the LLM prompts implementing the data selection, the crossover, and the mutation operators, which can perhaps be themselves optimized through the *CoEvo* evolutionary strategy.

Limitations and Ethical Considerations

Despite the interesting results, our study also has several important limitations. For instance, although the proposed framework is capable of finding better prompts automatically, reducing the human effort involved in creating and validating effective instructions, it is still not clear why these prompts work better. Further research into the prompting of large-scale language models can perhaps help us to better understand what kinds of prompts work, and how to design optimal prompts in a more principled manner.

The proposed approach also has limitations in terms of computational performance, since the method requires many calls to the language model being employed, sometimes with long prompts, thus involving significant costs (i.e., we require significant computational resources when using the proposed approach with open-source language models, as reported on the paper, or financial resources when using the proposed approach with commercial LLMs available through APIs).

Finally, the proposed approach was only tested on four benchmark tasks, and using a single LLM. While the four tasks involve different skills and domains, the considered set is by no means exhaustive. Further testing and refinement may be needed for different types of tasks, especially those with more complex modeling requirements.

We also note that we used GitHub Copilot during the development of our research work, and we used ChatGPT for minor verifications during the preparation of this manuscript.

References

Asma Ben Abacha, Wen-wai Yim, Griffin Adams, Neal Snider, and Meliha Yetisgen-Yildiz. 2023. Overview of the mediqa-chat 2023 shared tasks on the summarization & generation of doctor-patient conversations. In *Proceedings of the Clinical Natural Language Processing Workshop*.

Sondos Mahmoud Bsharat, Aidar Myrzakhan, and Zhiqiang Shen. 2023. Principled instructions are all you need for questioning Llama-1/2, GPT-3.5/4. *arXiv preprint arXiv:2312.16171*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Eldan Cohen and Christopher Beck. 2019. Empirical analysis of beam search performance degradation in neural sequence models. In *Proceedings of the 36th International Conference on Machine Learning*.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2023. PromptBreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*.

John Giorgi, Augustin Toma, Ronald Xie, Sondra Chen, Kevin An, Grace Zheng, and Bo Wang. 2023. WangLab at MEDIQA-chat 2023: Clinical note generation from doctor-patient conversations using large language models. In *Proceedings of the 5th Clinical Natural Language Processing Workshop*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2023. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *arXiv preprint arXiv:2309.08532*.

Tingfeng Hui, Lulu Zhao, Guanting Dong, Yaqi Zhang, Hua Zhou, and Sen Su. 2024. Smaller language models are better instruction evolvers. *arXiv preprint arXiv:2412.11231*.

Maël Jullien, Marco Valentino, Hannah Frost, Paul O'Regan, Donal Landers, and André Freitas. 2023. NLI4CT: Multi-evidence natural language inference for clinical trial reports. *arXiv preprint arXiv:2305.03598*.

Yuta Koreeda and Christopher D Manning. 2021. ContractNLI: A dataset for document-level natural language inference for contracts. *arXiv preprint arXiv:2110.01799*.

Laura Manor and Junyi Jessy Li. 2019. Plain english summarization of contracts. *arXiv preprint arXiv:1906.00424*.

OpenAI, Achiam J, et al. 2024. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. *arXiv preprint arXiv:2203.07281*.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with gradient descent and beam search. *arXiv preprint arXiv:2305.03495*.

Rohan Anil, Andrew M. Dai, et al. 2023. PaLM 2 technical report. *arXiv preprint arXiv:2305.10403*.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H. Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. 2024. How to train data-efficient LLMs. *arXiv preprint arXiv:2402.09668*.

Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with ChatGPT. *arXiv preprint arXiv:2302.11382*.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Yanggang Wang, Haiyu Li, and Zhilin Yang. 2022. GPS: Genetic prompt search for efficient few-shot learning. *arXiv preprint arXiv:2210.17041*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Wen-wai Yim, Yujuan Fu, Asma Ben Abacha, Neal Snider, Thomas Lin, and Meliha Yetisgen. 2023. ACI-BENCH: a novel ambient clinical intelligence dataset for benchmarking automatic visit note generation. *Nature Scientific Data*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Prompting Strategies

In Section 4.3 we described the use of several prompting strategies in our experiments. In this appendix, we detail these prompting strategies, by showing the structure and the individual components that form them. We also provide examples of the hand-crafted initial component descriptions that were used in our experiments. The appendix is organized into subsections, one for each of the considered tasks

Upon paper acceptance, all the initial descriptions that we used will be made available.

A.1 NLI4CT

For the NLI4CT task, we explored 2 different strategies. Figure 3 shows an example of a 0-shot prompt in this task. To ensure a *YES* or *NO* answer, we employ constrained decoding, looking at the probability of generating either token.

A.1.1 Zero-shot Prompting Strategy

The 0-shot strategy corresponds to the most straightforward implementation, presenting the single or the two Clinical Trial Report (CTR) sections as the premise, and the statement as the hypothesis. It uses 4 components: (1) describing the NLI4CT task, (2) describing the CTRs, (3) describing the statement, and (4) describing the answer. The structure of this strategy is shown in Figure 1.

We also show concrete examples of the prompts employed for each of the components, namely:

System: Assume the role of an automated system for the processing of domain-specific documentation, such as clinical or legal documents. The accuracy, robustness, consistency, and faithfulness of the reasoning performed by the system is critical in this context, and it is important to carefully consider the domain-specific terminology, to handle linguistic constructs such as temporal associations or negations, and to have robustness to different writing styles and vocabularies.

User: Examine the accuracy of clinical findings in Clinical Trial Reports (CTRs), which detail the effectiveness and safety of new treatments. CTRs are divided into four main parts: (1) Patient Selection Criteria, (2) Treatment Details, (3) Participant Data and Outcomes, and (4) Reported Side Effects. Verify the correctness of statements related to these sections, focusing on a single CTR or comparing two.

Consult the Clinical Trial Report (CTR) sections for comprehensive descriptions.

Primary Trial
"Adverse Events 1:
Total: 128/425 (30.12%)
Febrile neutropenia *2/425 (0.47%)
Anaemia *2/425 (0.47%)
Pancytopenia *2/425 (0.47%)
Coagulopathy *1/425 (0.24%)
Idiopathic thrombocytopenic purpura *0/425 (0.00%)
Microangiopathic haemolytic anaemia *1/425 (0.24%)
Neutropenia *0/425 (0.00%)
Pericardial effusion *1/425 (0.24%)
Acute coronary syndrome *1/425 (0.24%)
Adverse Events 2:
Total: 129/406 (31.77%)
Febrile neutropenia *6/406 (1.48%)
Anaemia *0/406 (0.00%)
Pancytopenia *0/406 (0.00%)
Coagulopathy *0/406 (0.00%)
Idiopathic thrombocytopenic purpura *1/406 (0.25%)
Microangiopathic haemolytic anaemia *0/406 (0.00%)
Neutropenia *1/406 (0.25%)
Pericardial effusion *1/406 (0.25%)
Acute coronary syndrome *0/406 (0.00%)"

Contemplate the ensuing statement formulated by a clinical expert or researcher. Carefully consider the conditions that should be present or absent from the CTR descriptions, when assessing the statement, together with associated details such as numeric quantities and other qualifiers.

"There less than 1% of either cohort of the primary trial was effect by Pancytopenia, but just over 5% of cohort 1 patients suffered from Coagulopathy."

Evaluate the given statement for its validity within the context of the Clinical Trial Report (CTR) descriptions. If the statement aligns with and is corroborated by the information presented in the CTR, respond with "YES". Conversely, if the statement contradicts or lacks support from the CTR descriptions, your response should be "NO".

Assistant: ANSWER:

Figure 3: Example of a 0-shot prompt in the NLI4CT dataset, using data instance `fcb195de-2143-44d8-8c46-136104554e2d`.

- Figure 4 shows two examples of prompts for the component corresponding to the NLI4CT *Task Description*;
- Figure 5 shows five examples of prompts for the component corresponding to the *Premise Description*;
- Figure 6 shows five examples of prompts for the component corresponding to the *Statement Description*;
- Figure 7 shows five examples of prompts for the component corresponding to the *Statement Description*;

A.1.2 Highlights and Self-Reasoning (HR) Prompting Strategy

In this setting, we highlight the most relevant information from the CTRs from the instance, based on the statement, and we use a two-step prompting strategy. Thus, this prompting strategy expands on the 0-shot strategy from Appendix A.1.1, by adding one component where we provide two sentences retrieved from the CTR, emphasizing the most relevant parts, plus two components specific to the use of reasoning.

The highlights are retrieved from the CTRs via dense embeddings generated with the model `Alibaba-NLP/gte-large-en-v1.5`[3]. We selected this model for its good balance between efficiency and retrieval performance. For each instance in the NLI4CT dataset, we take the parts of the CTR (separated by "." or "\n") and the statement. Then, we select the two most similar sentences from the CTR, by comparing the sentence embeddings with the embedding for the statement.

In addition to retrieval, we use a two-step prompting strategy. In the first step, we prompt the model for reasons why the given statement should be entailed or contradicted by the CTRs. Then, using this reasoning chain, we prompt the model for the final answer. The second step involves prompting the language model for a direct answer like in the 0-shot setting (Appendix A.1.1), but in this case we use an additional component providing the reasoning chain obtained from the first step. Figure 8 shows the prompt structure used for the second step, and Figure 9 illustrates some examples generated for the reasoning chain descriptions.

[3] https://huggingface.co/Alibaba-NLP/gte-large-en-v1.5

> Consider the task of determining semantic entailment relations between individual sections of Clinical Trial Reports (CTRs) and statements made by clinical domain experts. Note that CTRs outline the methodology and findings of a clinical trial, which are conducted to assess the effectiveness and safety of new treatments. Each trial involves 1-2 patient groups, called cohorts or arms, and these groups may receive different treatments, or have different baseline characteristics. The complete CTRs contain 4 sections, corresponding to (1) a list of the ELIGIBILITY CRITERIA corresponding to the conditions for patients to be allowed to take part in the clinical trial, (2) a description for the INTERVENTION that specifies the type, dosage, frequency, and duration of treatments being studied, (3) a summary of the RESULTS, detailing aspects such as the number of participants in the trial, the outcome measures, the units, and the conclusions, and (4) a list of ADVERSE EVENTS corresponding to signs and symptoms observed in patients during the clinical trial. In turn, the statements are sentences that make some type of claim about the information contained in one of the aforementioned sections, either considering a single CTR or comparing two CTRs. In order for the entailment relationship to be established, the claim in the statement should be related to the clinical trial information, it should be supported by the CTR, and it must not contradict the provided descriptions.

> Your task is determining support relationships between individual sections of Clinical Trial Reports (CTRs) and clinical statements. CTRs detail the methodology and findings of clinical trials, assessing effectiveness and safety of new treatments. CTRs consist of 4 sections: (1) ELIGIBILITY CRITERIA listing conditions for patient participation, (2) INTERVENTION description specifying type, dosage, frequency, and duration of treatments, (3) RESULTS summary detailing participants, outcome measures, units, and conclusions, and (4) ADVERSE EVENTS listing signs and symptoms observed. Statements make claims about information in these sections, either for a single CTR or comparing two.

Figure 4: Two examples for the *Task Description* component associated to the NLI4CT dataset.

A.2 ContractNLI Task

In the ContractNLI task, we used a single prompting strategy similar to the strategy based on highlights described in Appendix A.1.2, but without the reasoning part and adapting the retrieval method to the ContractNLI dataset.

In this task, the highlights can either be provided by an oracle or instead obtained through retrieval (using the same model described in Appendix A.1.2). Each NDA is divided into spans, which can range from a part of a sentence to a series of sentences. The dataset directly includes information about the spans relevant to a given statement

The following descriptions correspond to the information in the Clinical Trial Report (CTR) sections.

The descriptions provided next coincide with the content in a specific section of Clinical Trial Reports (CTRs), detailing relevant information to the trial.

The information provided next corresponds to the content found in one of the four standard clinical trial report sections.

Attend to the following descriptions pertaining to the contents found within one of the sections of Clinical Trial Reports (CTRs).

The descriptions that follow correspond to the information contained in one of the standard sections of clinical trial reports.

Figure 5: Five examples for the *Premise Description* component associated to the NLI4CT dataset.

(i.e., the oracle spans). However, to simulate a more realistic application scenario, we also considered an approach where we retrieve the spans, embedding every span from the NDA and the statement from the corresponding instance, and then selecting the k most similar spans that will be considered as the NDA's most relevant information. To determine the value of k, we chose the 90th percentile of the number of oracle spans across all the data instances, resulting in $k = 4$. Figure 10 shows the prompting structure used in this task. To constraint the possible generation to the "YES" or "NO" tokens, we also employed constrained decoding (similarly to what was described in Appendix A.1).

A.3 ToS-Sum Task on Plain English Summarization of Contracts

In the ToS-Sum task, we used a 1-shot prompting strategy, where we provided an example featuring a contract and the corresponding summary. To this end, we embed all the available contracts, and for each of them, we select the closest instance. Then, we take the corresponding contract and summary to be used as the example in the prompt. Retrieval is again made with the `Alibaba-NLP/gte-large-en-v1.5`[3] model, described in Appendix A.1.2.

Consider also the following statement generated by a clinical domain expert, a clinical trial organizer, or a medical researcher.

Contemplate the ensuing statement formulated by a clinical expert or researcher. Carefully consider the conditions that should be present or absent from the CTR descriptions, when assessing the statement, together with associated details such as numeric quantities and other qualifiers.

Review the subsequent statement provided by a domain specialist, attending to the medical terminology and carefully addressing any ambiguities.

Deliberate upon the subsequent statement formulated by an healthcare practitioner, a coordinator of clinical trials, or a medical researcher.

Reflect upon the ensuing statement crafted by a clinical expert.

Figure 6: Five examples for the *Statement Description* component associated to the NLI4CT dataset.

A.4 MEDIQA-CHAT Summarization Task

In the MEDIQA-CHAT dataset, we used a 1-shot prompting strategy. More specifically, we added an example note to help the model in assessing the appropriate structure for the clinical notes we want to generate. To this end, we embed all the available dialogues, and for each of them, we select the closest instance. Then, we take the corresponding clinical note to be used as an example in the prompt. Retrieval is again made with the `Alibaba-NLP/gte-large-en-v1.5`[3] model, described in Appendix A.1.2. Figure 12 shows the prompting structure used in this task.

B Mutation and Crossover Operations

The general prompt structure used in the implementation of the mutation and crossover operations are shown in Figures 13 and 17, respectively. The five concrete textual descriptions employed in these operations are also given next. In particular, regarding the mutation prompt:

- Figure 14 shows the different alternatives for the "Mutation Operation Description";
- Figure 15 shows the different alternatives for the "Instruction Description";

Answer YES or NO to the question of whether one can conclude the validity of the statement with basis on the Clinical Trial Report (CTR) information, carefully assessing the mentions in the statement that should be present or absent from the CTR descriptions.

Indicate with either YES or NO whether the statement is valid based on the Clinical Trial Report (CTR) descriptions. An answer of YES means that the statement is supported by the CTR descriptions, not contradicting the provided information.

Provide a YES or NO response indicating the statement's validity based on the information presented in the CTR descriptions. Do this by interpreting the medical terminology and the context in both the report and the statement, carefully assessing numeric quantities and other qualifiers, and addressing any ambiguities or gaps in the provided information.

You MUST respond with either YES or NO to indicate if the statement is valid based on the Clinical Trial Report (CTR) information, having determined that the statement is supported by the CTR data, and that it doesn't contradict the provided descriptions.

Check the statement's validity based on the clinical trial report data, providing a YES or NO response.

Figure 7: Five examples for the *Answer Description* component associated to the NLI4CT dataset.

- Figure 16 shows the different alternatives for the "Answer Description".

Regarding the crossover operation, Figure 18 shows three examples of the different alternatives for the "Crossover Operation Description". The remaining components are similar to the ones for the mutation prompt.

C Data Quality Assessment

We now describe the method used to filter the instances in the datasets. Specifically, to obtain a data quality score for all the examples in each of the 4 datasets used in our experiments, we developed a data quality estimation method based on the *Ask-LLM* approach proposed by Sachdeva et al. (2024). *Ask-LLM* prompts an instruction-tuned language model to obtain a data quality estimation for the examples of an unlabeled dataset. Then, data for model training can be selected based on this score.

Our goal was to have a general data quality estimation method that could be applied to the four datasets that were considered for the evaluation. With this in mind, we developed the prompt in Figure 19, which asked the model to identify which instances are the most uninformative. We used it to prompt the `Llama-3B-Instruct`[1] language model taking the probability of the tokens corresponding to positive answers (e.g., *yes*, *YES*, *Yes*, *Positive*, *positive*, and *POSITIVE*). Then, we select the instances with the lowest score. We also experimented with posing the question in the opposite fashion, followed by selecting the instances with the highest score or by taking the probability of a negative answer being generated. However, preliminary experiments favored the initially described version.

For the NLI datasets, we use the premise, the hypothesis, and the label to instantiate the "data instance" in the data quality prompt shown in Figure 19. For the summarization datasets, we add the original text and the target summary.

1 - Task Description
2 - Premise Description
`PRIMARY CLINICAL TRIAL REPORT`
`(SECONDARY CLINICAL TRIAL REPORT)`
3 - Statement Description
`CLINICAL STATEMENT`
4 - Highlights Description
`PRIMARY CLINICAL TRIAL REPORT HIGHLIGHTS`
`(SECONDARY CLINICAL TRIAL REPORT HIGHLIGHTS)`
5 - Reasoning Chain Description
`REASONING CHAIN`
6 - Answer Description

Figure 8: Prompt structure used in the second step of the self-reasoning prompting strategy for the NLI4CT dataset. The numbered parts correspond to the 5 components that are used in this strategy. Elements shown in a monospaced font are sample-dependent.

D Evolutionary Plots for *CoEvo* and `MC`

In this appendix, we show the plots of the evolutionary runs for the experiments reported in Section 4.3, using *CoEvo* and `MC`:

- Figure 20: `MC` (o-shot) in NLI4CT using 25% of the data selected at random;
- Figure 21: *CoEvo* (HR) in NLI4CT using 25% of the data selected at random;
- Figure 22: *CoEvo* (HR) in NLI4CT using 25% of the data selected by data quality;

Taking into account the statement and the Clinical Trial Report (CTR) descriptions, summarize in a very concise way the main reasoning steps that would allow one to conclude if the statement is supported by the CTR descriptions.

Considering the statement alongside the CTR descriptions, outline the critical reasoning steps necessary to determine whether the statement is corroborated by the information within the CTR. Keep it short.

After reviewing the statement and the Clinical Trial Report (CTR) descriptions, detail the logical sequence of reasoning steps required to assess whether the evidence in the CTR supports the statement.

In light of the given statement and the clinical trial information, enumerate the key reasoning processes needed to evaluate if the statement is valid. Be succinct.

Figure 9: Four examples for the *Reasoning Chain Description* component associated to the NLI4CT dataset.

1 - Task Description
2 - Premise Description
`NON DISCLOSURE AGREEMENT (NDA)`
3 - Statement Description
`STATEMENT`
4 - Highlights Description
`HIGHLIGHTS (ORACLE or RETRIEVED)`
5 - Answer Description

Figure 10: Prompt structure used in the ContractNLI task. Numbered parts correspond to the 4 prompt components and are sample-independent. Elements shown in a monospaced font are sample-dependent.

- Figure 24: `MC` (Oracle) in ContractNLI using 25% of the data selected at random;
- Figure 25: *CoEvo* (Oracle) in ContractNLI using 25% of the data selected at random;
- Figure 26: *CoEvo* (Retrieved) in ContractNLI using 25% of the data selected at random;
- Figure 27: *CoEvo* (Oracle) in ContractNLI using 25% of the data selected by data quality;
- Figure 28: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected at random;
- Figure 29: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected at random;

1 - Task Description
2 - Terms of Service Snippet Description
`Terms of Service SNIPPET`
3 - Example Description
`EXAMPLE SUMMARY`
4 - Answer Description

Figure 11: Prompt structure used in ToS-Sum task. Numbered parts correspond to the 4 prompt components used in this version of the task and are sample-independent. Elements shown in a monospaced font are sample-dependent.

1 - Task Description
2 - Dialogue Description
`DOCTOR-PATIENT DIALOGUE`
3 - Example Note Description
`EXAMPLE NOTE`
4 - Answer Description

Figure 12: Prompt structure used in the MEDIQA-CHAT Summarization task. Numbered parts correspond to the 4 prompt components and are sample-independent. Elements shown in a monospaced font are sample-dependent.

- Figure 30: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected by data quality;
- Figure 31: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected at random;
- Figure 32: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected at random;
- Figure 33: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected by data quality;

Each plot shows how the fitness function evolved over the different *CoEvo* and `MC` iterations.

E Final Optimized Prompts

In this appendix, we show the optimized prompts obtained for the experiments reported in Section 4.3:

- Table 5: `MC` (0-shot) in NLI4CT using 25% of the data selected at random;
- Table 6: *CoEvo* (0-shot) in NLI4CT using 25% of the data selected at random;
- Table 7: `MC` (Oracle) in ContractNLI using 25% of the data selected at random;

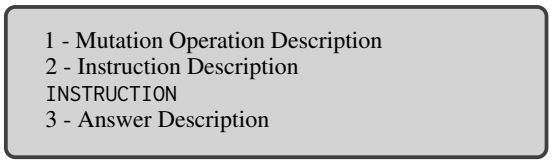

Figure 13: Prompt structure used in the mutation operation. Numbered parts correspond to the 3 prompt components used for the operation. INSTRUCTION is the string to which the operation is applied

- Table 8: *CoEvo* (Oracle) in ContractNLI using 25% of the data selected at random;
- Table 9: *CoEvo* (Retrieved) in ContractNLI using 25% of the data selected at random;
- Table 10: *CoEvo* (Oracle) in ContractNLI using 25% of the data selected by data quality;
- Table 11: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected at random;
- Table 12: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected at random;
- Table 13: *CoEvo* (1-shot) in ToS-Sum using 25% of the data selected by data quality;
- Table 14: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected at random;
- Table 15: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected at random;
- Table 16: *CoEvo* (1-shot) in MEDIQA-CHAT using 25% of the data selected by data quality;

F Hyper-Parameter Optimization

As detailed in Section 3.1, the numeric hyper-parameters used in the *CoEvo* algorithm are the following: population size (N), elite population size (E), mutation probability (p_m), crossover probability (p_c), and sampling temperature (S_T).

Early experiments with N ranging from 5 to 25 showed the advantages of a larger population. This is intuitive because a larger population allows the exploration of more combinations between the components. However, this also means we have more individuals to evaluate, making the algorithm computationally more demanding. To keep the algorithm relatively efficient, we set $N = 25$.

For E, we opted for a value corresponding to 20% of the total population. This value is larger than usual for evolutionary algorithms, but we obtained better results with this specific value in preliminary experiments.

Ideally, a grid search across all of the parameters would have been performed. However, for the sake of efficiency, only p_m, p_c, and S_T were tuned using grid search, while setting $N = 5$ and $E = 1$. The grid search considered three values for each of the aforementioned hyper-parameters. For the probabilities, we used $[0.25, 0.5, 0.75]$. For S_T, we used $[1.0, 5.0, 10.0]$.

Table 17 shows the results of the grid search, which were performed on the NLI4CT task. Even though we observed some variations in the obtained results, we conclude that settings with higher sampling temperature (more exploratory sampling) and a mutation probability in the lower range yield better results.

To ablate the relevance of mutation and crossover operations, we ran a configuration where we set the probabilities for these operations to 0, with an intermediate value on the remaining hyper-parameters. We concluded that it is beneficial to have both operations.

After this assessment, all the experiments with *CoEvo* use $p_m = p_c = 0.25$ and $S_T = 10$.

G Looking at an Experiment in Depth

As a way to make it possible to assess the evolutionary path of prompts during the execution of *CoEvo* we track the history of every component at every iteration. This allows us to trace back the family tree of the components of the final optimized prompt. In this appendix, we illustrate these results for the *CoEvo* (0-shot) experiment in the NLI4CT dataset using 25% of the available data. We show the evolutionary path for each component of the optimized prompt in this experiment:

- Table 18 presents results for the *Task Description* component;
- Table 19 presents results for the *CTR Description* component;
- Table 20 presents results for the *Statement Description* component;
- Table 21 presents results for the *Answer Description* component.

The first row of each table shows the first component description generated through crossover

	Optimized Description
Task Desc.	Your task is determining the support of clinical statements from individual sections of Clinical Trial Reports (CTRs). CTRs detail the methodology and findings of clinical trials, assessing effectiveness and safety of new treatments. CTRs consist of 4 sections: (1) ELIGIBILITY CRITERIA listing conditions for patient participation, (2) INTERVENTION description specifying type, dosage, frequency, and duration of treatments, (3) RESULTS summary detailing participants, outcome measures, units, and conclusions, and (4) ADVERSE EVENTS listing unexpected signs and symptoms. Statements make claims about information in these sections, either for a single CTR or comparing two.
CTR Desc.	The descriptions provided next coincide with the content in a specific section of Clinical Trial Reports (CTRs), detailing relevant information to the trial.
Statement Desc.	Contemplate the ensuing statement formulated by a clinical expert or researcher. Carefully consider the conditions that should be present or absent from the CTR descriptions, when assessing the statement, together with associated details such as numeric quantities and other qualifiers.
Answer Desc.	Indicate with either YES or NO whether the statement is valid based on the Clinical Trial Report (CTR) descriptions. An answer of YES means that the statement is supported by the CTR descriptions, not contradicting the provided information.

Table 5: Optimized components for the NLI4CT MC (0-shot) experiment, using 25% of the available data selected at random (reported in Table 1).

	Optimized Description
Task Desc.	Evaluate the validity of medical claims based on the content found in individual parts of Clinical Trial Reports (CTRs). These reports provide insights into the efficacy and safety of experimental therapies. A typical CTR is divided into four segments: (1) PARTICIPANT CRITERIA outlining eligibility for trial enrollment, (2) THERAPY DETAILS describing the intervention's nature, dosage, frequency, and length, (3) STUDY RESULTS summarizing participant demographics, outcome metrics, measurement units, and inferences, and (4) SIDE EFFECTS documenting any unforeseen adverse reactions. Assess whether statements are accurate reflections of data from a single CTR or a comparison between two.
CTR Desc.	Synthesize the essential information from the provided clinical trial report sections, ensuring clarity and confidence for precise task execution.
Statement Desc.	Review the subsequent statement provided by a domain specialist, attending to the medical terminology and carefully addressing any ambiguities.
Answer Desc.	Examine the Clinical Trial Report (CTR) to verify the accuracy of a statement by identifying relevant elements within the document.

Table 6: Optimized components for the NLI4CT *CoEvo* (0-shot) experiment, using 25% of the available data selected at random (reported in Table 1).

and/or mutation during the optimization path (i.e., resulting in a description different from the initial hand-crafted version). The remaining rows show different prompt descriptions associated with individuals that further improved the best fitness score.

	Optimized Description
Task Desc.	Analyze the semantic entailment between segments of a Non-Disclosure Agreement and statements regarding its confidentiality clauses.
Doc. Desc.	Examine the full NDA document.
Statement Desc.	Examine the commentary from a legal authority, contract overseer, or regulatory advisor.
Hihglights Desc.	Given that the complete Non-Disclosure Agreement (NDA) is quite extensive, attend to the following summary of the most important legal information.
Answer Desc.	Determine if the Non-Disclosure Agreement (NDA) supports the statement's validity.

Table 7: Optimized components for the ContractNLI MC (Oracle) experiment, using 25% of the available data selected at random (reported in Table 2).

	Optimized Description
Task Desc.	Conduct a comprehensive review of Non-Disclosure Agreements (NDAs), focusing on the assessment of confidentiality clauses, data protection measures, the scope of confidentiality, and the responsibilities of the parties involved.
Doc. Desc.	Comprehend the lifespan of an NDA's enforceability and the repercussions of violating its stipulations. Strive for a thorough grasp of the Non-Disclosure Agreement.
Statement Desc.	Convene a team comprising legal advisors, contract managers, compliance specialists, expert counsellors, and contract law specialists to meticulously craft a comprehensive manual. This manual should serve as a definitive resource for adhering to legal requirements and optimizing the oversight of contractual engagements.
Highlights Desc.	Conduct a thorough analysis of the Non-Disclosure Agreement (NDA) document, focusing specifically on the identification and understanding of its key legal elements.
Answer Desc.	Determine the truthfulness of the provided statement by evaluating its correctness.

Table 8: Optimized components for the ContractNLI *CoEvo* (Oracle) experiment, using 25% of the available data selected at random (reported in Table 2).

	Optimized Description
Task Desc.	Conduct a thorough examination of the Non-Disclosure Agreements (NDAs) to verify that the confidentiality clauses are explicitly supported by the content of the agreements. This includes a detailed assessment of how information is defined, the delineation of responsibilities, the specified duration of the confidentiality obligation, and the articulation of any exceptions to the confidentiality terms.
Doc. Desc.	Review the entire Non-Disclosure Agreement (NDA).
Statement Desc.	Listen to the following statement from a contract law expert.
Highlights Desc.	Perform an in-depth examination of the Non-Disclosure Agreement (NDA) to pinpoint critical confidentiality stipulations, outline the distinct responsibilities assigned to each party, and define the NDA's duration and geographic scope. Compile a comprehensive and precise summary that encapsulates the fundamental confidentiality provisions, delineates each party's individual obligations, and elucidates the NDA's validity period and territorial application, crafted for industry experts.
Answer Desc.	Assess the accuracy of the legal assertion and reply with either 'ACCURATE' or 'INACCURATE'.

Table 9: Optimized components for the ContractNLI *CoEvo* (Retrieved) experiment, using 25% of the available data selected at random (reported in Table 2).

	Optimized Description
Task Desc.	-
Doc. Desc.	Conduct a thorough examination of the Confidentiality Agreement, pinpointing and comprehending the distinct confidentiality duties and constraints applicable to each participant, with an emphasis on recognizing and upholding your obligation to preserve confidentiality as stipulated in the document.
Statement Desc.	Assess the findings of a legal advisor, a contract manager, or a compliance specialist. Provide a clear and detailed explanation of their analyses and conclusions to ensure a thorough comprehension of the results.
Highlights Desc.	Given that the complete Non-Disclosure Agreement (NDA) is quite extensive, attend to the following summary of the most important legal information.
Answer Desc.	Assess whether the given statement adheres to the stipulations outlined in the Non-Disclosure Agreement (NDA) and confirm compliance by stating "YES" if it aligns with the agreement's requirements.

Table 10: Components associated to the best result obtained for the ContractNLI dataset. These correspond to the *CoEvo* (Oracle) experiment, using 25% of the available data selected by data quality (reported in Table 2).

	Optimized Description
Task Desc.	Consider the task of writing a summary of portion of a Terms of Services (TOS) Contract. Terms of Service (TOS) are the legal agreements between a service provider and a person who wants to use that service. The person must agree to abide by the terms of service in order to use the offered service. Your goal is to write a very small summary of the portion of the TOS you will be provided with. The summary should be concise and easy to read and understand by anyone.
ToS snippet Desc.	The following section is an excerpt from the Terms of Service Agreement, outlining specific conditions and guidelines that apply to the use of the service.
Answer Desc.	Now, generate a very concise summary of the provided TOS section using straightforward language. No additional explanations or extra information should be included. Focus on key points like user obligations, service limitations, and important restrictions. Ensure the summary is brief and easy to understand.

Table 11: Optimized components for the ToS-Sum MC (1-shot) experiment, using 25% of the available data selected at random (reported in Table 3).

	Optimized Description
Task Desc.	You are tasked with summarizing a section of a Terms of Services (TOS) legal agreement. Your goal is to write a very minmal and easy to read summary of the portion of the TOS you will be provided with.
ToS snippet Desc.	The following section is an excerpt from the Terms of Service Agreement, outlining specific conditions and guidelines that apply to the use of the service.
Example Desc.	Below you have an example of a summary made for a different Terms of Conditions text.
Answer Desc.	Condense the Terms of Service (TOS) section into a clear, concise summary of 1 or 2 sentences using plain language. Highlight the essential obligations of the user, the scope of the service, and any critical restrictions or penalties, strictly adhering to the original TOS content.

Table 12: Optimized components for the ToS-Sum *CoEvo* (1-shot) experiment, using 25% of the available data selected at random (reported in Table 3).

	Optimized Description
Task Desc.	Craft a brief overview of a specific section from a Service Agreement (SA). A Service Agreement (SA) outlines the legal terms between a service provider and a user. To access the service, the user must consent to the SA's conditions. Your objective is to compose a succinct summary of the SA segment you will receive. Ensure the summary is clear and straightforward.
ToS snippet Desc.	I will provide you with a specific section from the TOS agreement. Carefully read and understand the section, focusing on important aspects such as user rights, obligations, privacy policies, data sharing, and any key restrictions or terms that users should be aware of. Identify the most critical points and anything that could impact how users interact with the service.
Example Desc.	Craft a concise, user-friendly summary that captures the essence of the provided demonstration and the Terms of Service (ToS) document. Ensure the language is clear, professional, and devoid of complex legal terminology. This summary should serve as a comprehensive guide, empowering individuals to understand and adhere to the guidelines with confidence.
Answer Desc.	Now, generate a very concise summary of the provided TOS section using straightforward language. No additional explanations or extra information should be included. Focus on key points like user obligations, service limitations, and important restrictions. Ensure the summary is brief and easy to understand.

Table 13: Components associated to the best result obtained for the ToS-Sum dataset. These correspond to the *CoEvo* (1-shot) experiment, using 25% of the available data selected by data quality (reported in Table 3).

	Optimized Description
Task Desc.	Synthesize a concise medical report from the dialogue between the patient and doctor, capturing all pertinent medications, diagnoses, health issues, injuries, and symptoms. Organize the report using either SOAP or APSO format for enhanced clarity and succinctness.
Dialogue Desc.	Examine the following dialogue between a doctor and a patient.
Example Desc.	Examine the clinical note's architecture.
Answer Desc.	Compose a comprehensive clinical note with the following headings: Assessment, Assessment and Plan, Clinical History, Chief Complaint, Current Medications, Examination, Family History, History of Present Illness, Impression, Instructions, Medical History, Medications, Past Medical History, Past Surgical History, Physical Exam, Physical Examination, Plan, Results, Review of Systems, Social History, Vital Signs, Vital Signs Reviewed.

Table 14: Optimized components for the MEDIQA-CHAT MC (1-shot) experiment, using 25% of the available data selected at random (reported in Table 4).

	Optimized Description
Task Desc.	Compose a clinical note that begins with the patient's chief complaint, followed by a structured HPI, ROS, PE, and concludes with an A/P, ensuring adherence to the standard clinical documentation format.
Dialogue Desc.	Examine the conversation transcript involving a doctor and a patient inquiring for health guidance.
Example Desc.	Analyze the clinical notes from two distinct patient visits and the corresponding medical record illustration for a different consultation. Ensure a thorough examination of the organization and content specific to each case, comparing and contrasting the written documentation with the visual medical record representation.
Answer Desc.	Create a clinical note with each section filled with pertinent clinical information derived from the dialogue. Use the following section headings: ASSESSMENT, ASSESSMENT AND PLAN, CLINICAL HISTORY, CHIEF COMPLAINT, CURRENT MEDICATIONS, EXAM, FAMILY HISTORY, HISTORY OF PRESENT ILLNESS, IMPRESSION, INSTRUCTIONS, MEDICAL HISTORY, MEDICATIONS, PAST HISTORY, PAST SURGICAL HISTORY, PHYSICAL EXAM, PHYSICAL EXAMINATION, PLAN, RESULTS, REVIEW OF SYSTEMS, SOCIAL HISTORY, VITALS, or VITALS REVIEWED. Ensure all sections contain relevant data from the conversation.

Table 15: Optimized components for the MEDIQA-CHAT *CoEvo* (1-shot) experiment, using 25% of the available data selected at random (reported in Table 4).

	Optimized Description
Task Desc.	Compile a clinical note with sections on History of Present Illness, Physical Examination, Results, and Assessment and Plan, reflecting the dialogue's clinical details accurately.
Dialogue Desc.	Examine the conversation between a doctor and a patient.
Example Desc.	Consider the following example of a clinical note as a reference. When creating the new clinical note, mirror the format of the example.
Answer Desc.	Compile a detailed clinical note from the dialogues, following the HISTORY OF PRESENT ILLNESS, PHYSICAL EXAM, RESULTS, ASSESSMENT AND PLAN format, ensuring the content is accurate, clear, and concise for expert review.

Table 16: Components associated to the best result obtained for the MEDIQA-CHAT dataset. These correspond to the *CoEvo* (1-shot) experiment, using 25% of the available data selected by data quality (reported in Table 4).

S_T	p_M	p_C	Initial F1 (Dev)	Final F1 (Dev)	No. of Improvements	F1 (Test)	Faith. (Test)	Consis. (Test)
1.0	0.25	0.25	0.6327	0.6617	2	0.5771	0.7512	0.7365
1.0	0.25	0.50	0.6327	0.6493	1	0.5911	0.7558	0.7645
1.0	0.25	0.75	0.6327	0.6519	2	0.5981	0.7141	0.7176
1.0	0.50	0.25	0.6327	0.6511	4	0.5830	0.6551	0.7099
1.0	0.50	0.50	0.6327	0.6560	3	0.5785	0.6829	0.7125
1.0	0.50	0.75	0.6327	0.6581	3	0.5909	0.6667	0.7048
1.0	0.75	0.25	0.6327	0.6561	3	0.5743	0.728	0.7365
1.0	0.75	0.50	0.6327	0.6744	3	0.5881	0.7153	0.7454
1.0	0.75	0.75	0.6327	0.6682	4	0.5897	0.735	0.7546
5.0	0.25	0.25	0.6327	0.6727	2	0.5866	0.6667	0.7142
5.0	0.25	0.50	0.6327	0.6707	2	0.5971	0.6979	0.7427
5.0	0.25	0.75	0.6327	0.6743	2	0.5969	0.7523	0.7558
5.0	0.50	0.25	0.6327	0.6453	2	0.5735	0.6644	0.7123
5.0	0.50	0.50	0.6327	0.6452	1	0.5646	0.5833	0.7166
5.0	0.50	0.75	0.6327	0.6518	3	0.5831	0.6725	0.7205
5.0	0.75	0.25	0.6327	0.6814	5	0.5981	0.6875	0.7331
5.0	0.75	0.50	0.6327	0.6611	2	0.5792	0.6053	0.7205
5.0	0.75	0.5	0.6327	0.6346	2	0.5664	0.6829	0.7132
10.0	**0.25**	**0.25**	0.6327	0.6813	3	0.5982	0.6782	0.7137
10.0	0.25	0.50	0.6327	0.6526	2	0.5934	0.6007	0.7261
10.0	0.25	0.75	0.6327	0.6534	1	0.5920	0.7257	0.7461
10.0	0.50	0.25	0.6327	0.6592	2	0.5875	0.7234	0.7437
10.0	0.50	0.50	0.6327	0.6740	3	0.5980	0.6481	0.7316
10.0	0.50	0.75	0.6327	0.6327	2	0.5870	0.7211	0.7198
10.0	0.75	0.25	0.6327	0.6439	2	0.5488	0.8067	0.7737
10.0	0.75	0.50	0.6327	0.6618	3	0.5879	0.6609	0.7265
10.0	0.75	0.75	0.6327	0.6472	1	0.5815	0.6632	0.7079

Table 17: Results for a grid search experiment across 3 of the hyper-parameters associated to *CoEvo*: sampling temperature S_T, mutation probability p_M, and crossover probability p_C. The grid search was done using the 0-shot setting in the NLI4CT dataset, using 25% of the available data chosen at random. The columns named "Initial F1 (Dev)", "Final F1 (Dev)", and "No. of Improvements" refer to the evolutionary run (evaluated on the 25% of the data). The last three columns refer to the evaluation of the resulting optimized prompt on the test set.

Ite.	ID	Task Description	From	Cros.	Mut.
5	0	Your task is determining the support of clinical statements from individual sections of Clinical Trial Reports (CTRs). CTRs detail the methodology and findings of clinical trials, assessing effectiveness and safety of new treatments. CTRs consist of 4 sections: (1) ELIGIBILITY CRITERIA listing conditions for patient participation, (2) INTERVENTION description specifying type, dosage, frequency, and duration of treatments, (3) RESULTS summary detailing participants, outcome measures, units, and conclusions, and (4) ADVERSE EVENTS listing unexpected signs and symptoms. Statements make claims about information in these sections, either for a single CTR or comparing two.	-	-	-
6	3	Evaluate the validity of medical claims based on the content found in individual parts of Clinical Trial Reports (CTRs). These reports provide insights into the efficacy and safety of experimental therapies. A typical CTR is divided into four segments: (1) PARTICIPANT CRITERIA outlining eligibility for trial enrollment, (2) THERAPY DETAILS describing the intervention's nature, dosage, frequency, and length, (3) STUDY RESULTS summarizing participant demographics, outcome metrics, measurement units, and inferences, and (4) SIDE EFFECTS documenting any unforeseen adverse reactions. Assess whether statements are accurate reflections of data from a single CTR or a comparison between two.	5	-	2

Table 18: Evolution of the task description component obtained in the *CoEvo*$_{(0\text{-shot})}$ run using 25% of the available data chose at random. The first two columns (Iter. and ID) identify the component description by the iteration in which it was generated, together with the respective ID inside the iteration. Task description is the string corresponding to the component. The last three columns show how the string was generated. The column named From shows which components were used in its creation, while the column named Cros. indicates if the crossover operation was applied, and which of the 5 crossover operator prompts was used. The column named Mut. indicates if the mutation operation was applied, and which of the 5 mutation operator prompts was used. When only the mutation operation takes place, only one prompt is used in the generation. If both operations take place, the mutation is applied to the resulting prompt from the crossover operation.

Ite.	ID	CTR Description	From	Cros.	Mut.
13	0	The descriptions that follow correspond to the information contained in one of the standard sections of clinical trial reports.	-	-	-
13	1	The information provided next corresponds to the content found in one of the four standard clinical trial report sections.	-	-	-
14	4	Synthesize the essential information from the provided clinical trial report sections, ensuring clarity and confidence for precise task execution.	Iter 13, ID 0,1	1	2

Table 19: Evolution of the CTR description component obtained in the *CoEvo*$_{(0\text{-shot})}$ run using 25% of the available data chose at random. See the caption of Table 19 for details on the columns.

Ite.	ID	Statement Description	From	Cros.	Mut.
0	0	Review the subsequent statement provided by a domain specialist, attending to the medical terminology and carefully addressing any ambiguities.	-	-	-

Table 20: Evolution of the statement Description component obtained in the *CoEvo*$_{(0\text{-shot})}$ run using 25% of the available data chose at random. See the caption of Table 19 for details on the columns.

Ite.	ID	Answer Description	From	Cros.	Mut.
3	0	Answer YES or NO to the question of whether one can conclude the validity of the statement with basis on the Clinical Trial Report (CTR) information, carefully assessing the mentions in the statement that should be present or absent from the CTR descriptions.	-	-	-
13	0	Evaluate the presence or absence of specific elements within the Clinical Trial Report (CTR) to determine the validity of a given statement. Assess whether the statement accurately reflects the information provided in the CTR.	Iter 3, ID 0	-	4
14	4	Examine the Clinical Trial Report (CTR) to verify the accuracy of a statement by identifying relevant elements within the document.	Iter 13	-	1

Table 21: Evolution of the Answer Description component obtained in the *CoEvo*$_{(0\text{-shot})}$ run using 25% of the available data chose at random. See caption of Table 19 for details on the Columns.

Consider the problem of rewriting a textual instruction, in which the objective is to rephrase the descriptions by changing the phrasing or the ordering of the explanations, while keeping the exact same meaning. Assume that the audience for the resulting instruction consists of experts in the field. The rewritten instruction can either be shorter, summarizing the main points while keeping consistency with the original intent, or it can be made longer, by adding definitions and further clarifications. You will be penalized for the inclusion of incorrect information. The re-written instruction should be concise and direct, and it should inform the execution of the task in a clearer way than the original instruction.

Consider the problem of rephrasing a textual instruction, where the main objective is to rewrite the descriptions accurately. Focus on making the new instruction more concise, but it is crucial to retain the core information and essence of the original instruction. The end result should be a clear and direct rephrasing that maintains the intended meaning, without any loss of important details. The rephrased instruction should be easily comprehensible by experts in the field, ensuring no ambiguity or misinterpretation.

Reflect on the task of paraphrasing a written instruction, where the goal is to modify the expalnations by reordering or replacing words and phrases, while keeping the same overall meaning. Focus on generating a new instruction that, while being lexically distinct, remains simple, understandable, and clear.

You are tasked with modifying an input textual instruction, by changing the ordering and the level of detail in the associated explanations. The goal is to create a new instruction that not only preserves the essential information, but that also improves upon it, making it easier for experts in the field to execute the task accurately and efficiently.

Think about the task of modifying a written instruction, with a focus on improving the performance of AI systems when executing the task. You should rephrase the instruction by changing the wording and the ordering of the commands, ensuring that the new version enhances the clarity and effectiveness of the descriptions. Attend to the nuances of the original instruction, and make sure that the rewritten version is precise, actionable, and free of ambiguities, thereby facilitating a seamless execution of the task.

Figure 14: Alternatives for the *Mutation Operation Task Description* component.

Noting the detailed task description, attend to the instruction shown next in quotes.

Given the previous problem formulation, consider the quoted instruction given next. Pay attention to the core information, and reflect on how the phrasing or the ordering of the explanations could be changed.

Taking into account the previous explanation, review the following instruction shown in quotes.

Attend to the textual instruction shown next in quotes, reflecting on how to summarize the core information.

Following the previous task description, reflect on the quoted instruction. Identify the core information, and consider adding more details in the appropriate places, so that the resulting description is more comprehensive.

Figure 15: Alternatives for the *Mutation Operation Instruction Description* component.

Generate the rephrased instruction without any additional explanation, keeping it short and simple.

Considering all previous information, produce the resulting instruction, without giving any additional context or details about the reasoning process.

Generate the result of rewriting the instruction, carefully considering the task description and without explaining how the result was produced.

Produce only the resulting instruction, without any additional context.

State the result of rephrasing the instruction according to the task description, omitting any further contextualization.

Figure 16: Alternatives for the *Mutation Operation Answer Description* component.

1 - Crossover Operation Description
2 - Instructions Description
`INSTRUCTION 1`
`INSTRUCTION 2`
3 - Answer Description

Figure 17: Prompt structure used in the crossover operation. Numbered parts correspond to the 3 prompt components used for the operation. INSTRUCTION 1 and INSTRUCTION 2 are the strings to which the operation is applied.

Consider the problem of combining two different textual instructions, pertaining to the same task. The objective is to rephrase the main information common to the two descriptions, while keeping their meaning and intent. Assume that the audience for the resulting instruction consists of experts in the field. The combined instruction can either be shorter, summarizing the main points while keeping consistency with the original intent, or it can be made longer, by adding definitions and further clarifications. You will be penalized for the inclusion of incorrect information. The combined instruction should be concise and direct, and it should inform the execution of the task in a clearer way than the original instructions.

Think about the task of merging two different textual descriptions for the same problem. The goal is to rephrase the main information that is common to both descriptions, leveraging the best elements from each while preserving their meaning and intent. Your task is to generate a concise and direct instruction that communicates the necessary steps for task execution more clearly than the original versions. This requires careful consideration of the language and details used in both instructions, to ensure that the rephrased version is not only succinct, but also informative and easy to understand. The end result should be a streamlined instruction that experts in the field can follow with ease and confidence.

Consider how to combine two separate textual instructions for the same task. The primary objective is to rephrase the main information that is common to both descriptions, using different words and changing the ordering of the statements, while ensuring that their meaning and intent remain intact. It is important to include all relevant details from both instructions, regardless of whether this makes the final instruction longer. The resulting instruction should be clear, comprehensive, and informative, providing a thorough guide for task execution. This means paying attention to nuances and specific details in both texts, integrating them seamlessly to produce a unified directive that leaves no room for ambiguity. The final instruction should be detailed enough to cater to experts in the field, ensuring they have all the information needed to perform the task efficiently.

Figure 18: Three of the alternatives for the *Crossover Operation Task Description* component.

System: Assume the role of an automated system for the processing of domain-specific documentation, such as clinical or legal documents. The accuracy, robustness, consistency, and faithfulness of the reasoning performed by the system is critical in this context, and it is important to carefully consider the domain-specific terminology, to handle linguistic constructs such as temporal associations or negations, and to have robustness to different writing styles and vocabularies.

User: The following textual description corresponds to a particular instance from a dataset.

`data instance`

Consider the task of determining whether or not the instance is uninformative, in what regards exemplifying the contents of the dataset. Notice that an uninformative instance should be very easy to analyze and classify, failing to illustrate the particular challenges and the corner cases that may exist in the complete dataset to which it belongs. Its contents may also provide little or no useful information, likely failing to elicit a meaningful response from its analysis. Your goal is to assess whether the instance corresponds to an uninformative example that should be ignored, e.g. when assessing the performance of a large language model over the complete dataset. Taking into account the aforementioned goal, attend carefully to the contents of the instance.

`data instance`

Answer affirmatively if you deem the instance to be uninformative, or negatively otherwise.

Assistant: ANSWER

Figure 19: The prompt used to assess the data quality score for each data instance in each of the 4 datasets.

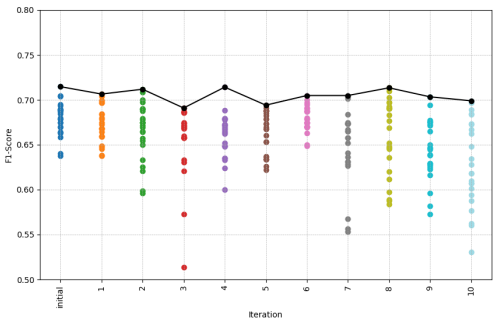

Figure 20: Evolutionary run for the NLI4CT `MC` (0-shot) experiment, using 25% of the available data selected at random (reported in Table 1).

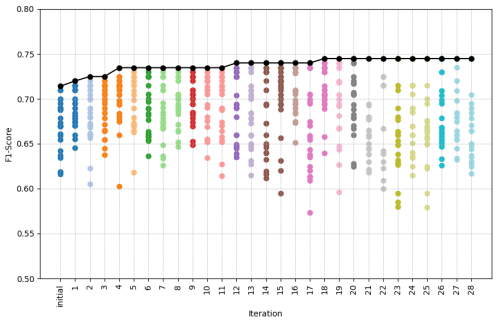

Figure 21: Evolutionary run for the NLI4CT *CoEvo* (Highlights) experiment, using 25% of the available data selected at random (reported in Table 1).

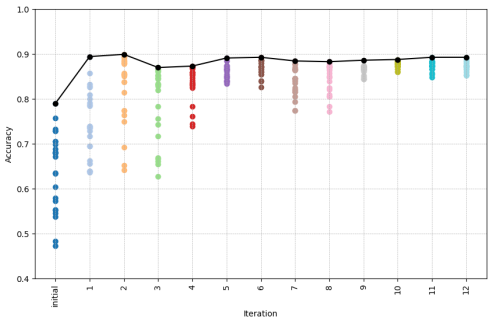

Figure 24: Evolutionary run for the ContractNLI `MC` (Oracle) experiment, using 25% of the available data selected at random (reported in Table 2).

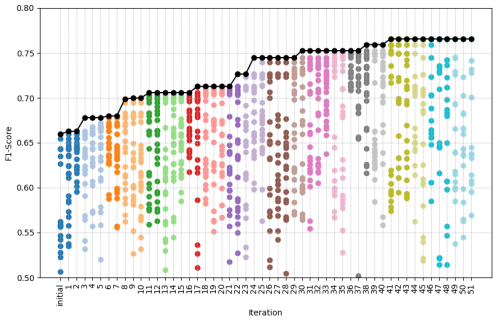

Figure 22: Evolutionary run for the NLI4CT *CoEvo* (Highlights) experiment, using 25% of the available data selected by data quality (reported in Table 1).

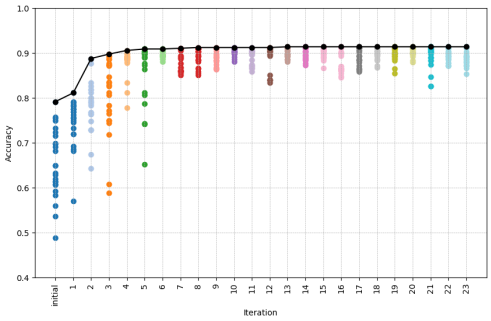

Figure 25: Evolutionary run for the ContractNLI *CoEvo* (Oracle) experiment, using 25% of the available data selected at random (reported in Table 2).

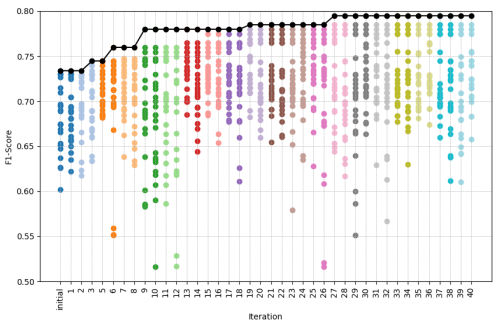

Figure 23: Evolutionary run for the NLI4CT *CoEvo* (Self Reas.) experiment, using 25% of the available data selected at random (reported in Table 1).

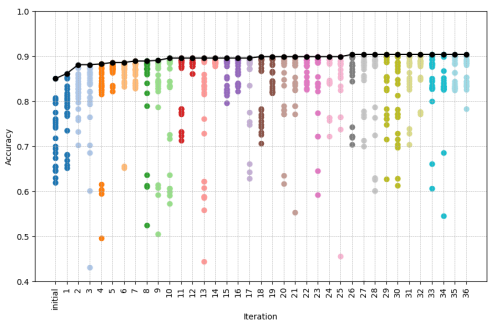

Figure 26: Evolutionary run for the ContractNLI *CoEvo* (Retrieved) experiment, using 25% of the available data selected at random (reported in Table 2).

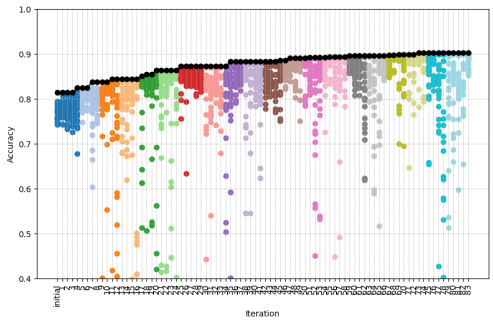

Figure 27: Evolutionary run for the ContractNLI *CoEvo* (Oracle) experiment, using 25% of the available data selected by data quality (reported in Table 2).

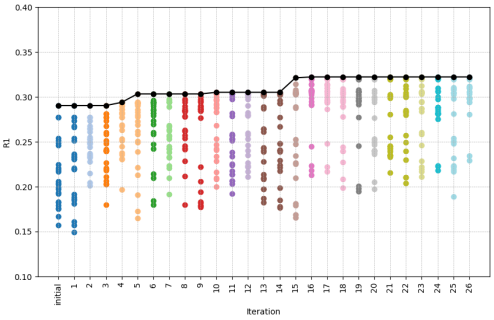

Figure 30: Evolutionary run for the ToS-Sum *CoEvo* (1-shot) experiment, using 25% of the available data selected by data quality (reported in Table 3).

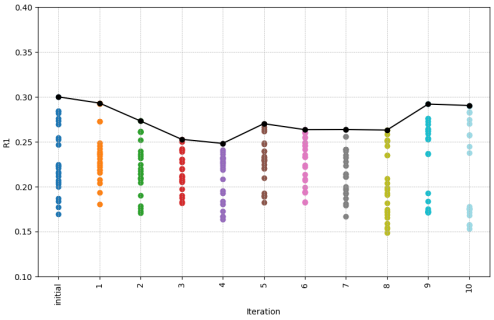

Figure 28: Evolutionary run for the ToS-Sum `MC` (1-shot) experiment, using 25% of the available data selected at random (reported in Table 3).

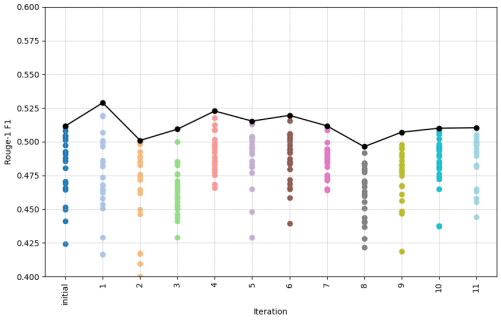

Figure 31: Evolutionary run for the MEDIQA-CHAT `MC` (1-shot) experiment, using 25% of the available data selected at random (reported in Table 4).

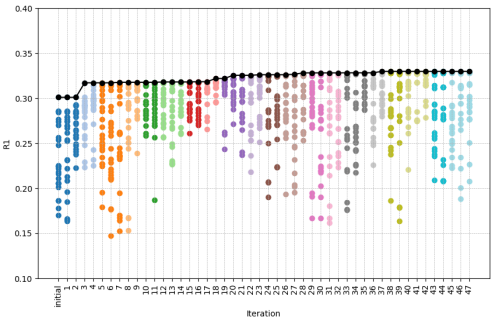

Figure 29: Evolutionary run for the ToS-Sum *CoEvo* (1-shot) experiment, using 25% of the available data selected at random (reported in Table 3).

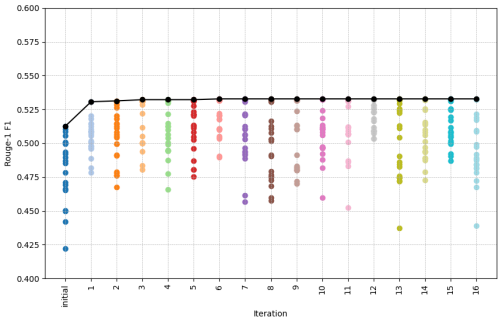

Figure 32: Evolutionary run for the MEDIQA-CHAT *CoEvo* (1-shot) experiment, using 25% of the available data selected at random (reported in Table 4).

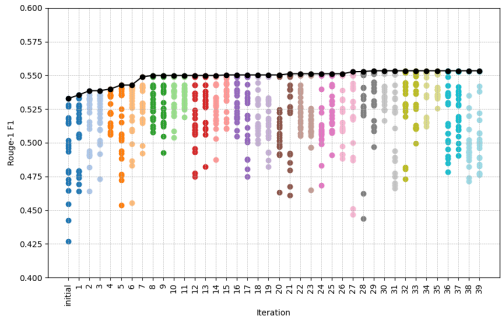

Figure 33: Evolutionary run for the MEDIQA-CHAT *CoEvo* (1-shot) experiment, using 25% of the available data selected by data quality (reported in Table 4).