Generating-Recovering Annotation: GPT Self-Supervision for an Efficient Data Annotator

Xiaohuan Pei, Yanxi Li, and Chang Xu, *Senior Member, IEEE*

***Abstract*—Annotating data poses a significant challenge across various domains, frequently requiring the allocation of significant time and specialized knowledge by human experts. Despite existing efforts to use large language models for annotation tasks, significant problems such as limited applicability to unlabeled data, the absence of self-supervised methods, and the lack of focus on complex structured data still persist. In this work, we propose a GPT self-supervision annotation method, *Generating-Recovering Annotation* (GRA), which embodies a generating-recovering paradigm that leverages the few shot learning capabilities of the Generative Pretrained Transformer (GPT). The proposed approach comprises a template optimization phase followed by a label generation phase. In the template optimization phase, we sample an unlabeled data from the candidate set as part of the prompt for GPT to generate a textual summary, which is then used to recover the original data. The alignment score between the recovered and original data serves as a self-supervision navigator to refine the process. In the generation stage, the optimally selected sample serves as a template in the prompt and is applied to generating summaries from challenging datasets. The annotation performance is evaluated by tuning several human feedback reward networks and by calculating alignment scores between original and recovered data at both sentence and structure levels. Our self-supervised annotation method consistently achieves competitive scores, convincingly demonstrating its robust strength in various annotation tasks.**

***Index Terms*—Data Annotation, Large Language Model, Few shot learning.**

I. Introduction

Large language models represented by Generative Pre-trained Transformer(GPT) family [1]–[3] have made breakthrough advancements in recent years, achieving state-of-the-art performance across various deep learning tasks. Among these tasks, data annotation is the fundamental and indispensable first step in the process of AI research, as it lays the groundwork by providing labeled data that serves as the foundation for training and evaluating models [4]–[7]. The quality of data label generation directly impacts all subsequent tasks, making it the cornerstone of the entire deep learning process [8], [9].

The annotation task poses the significant challenges. First, the complexity of the data makes it time-consuming for experts to annotate each sample [10]–[13]. For example, computational graphs provide valuable insights into the design of neural network structures and annotating these cells requires significant effort to identify and summarize similar subsequences as parallel blocks [14], which is nearly impossible to identify and isolate. Second, the exclusive use of subjective human-annotated summaries presents evaluation challenges [3], [15]. Employing these subjective summaries as supervised learning labels could lead to annotation tasks reflecting human bias, not the impartial correlation between the data and its summary.

Recent research has harnessed the capabilities of large language models for supervised data annotation. As revealed by [16]–[18], GPT models can annotate classification tasks through the direct design of diverse prompts. Furthermore, the study by [19] broadens the scope of this annotation task, enabling the generation of dialogues. Despite impressive progress, the problems of annotation task still remain. First, current studies are primarily centered around supervised annotation with human-labeled data, which restricts the annotation method's applicability to more general unlabeled data. In most scenarios, human-labeled data is unavailable to guide the annotation process. Second, prior research has been limited to the design of various prompts, without considering feedback mechanisms to refine specific parts of the prompts. Third, there's a lack of research focused on annotating complex structured data. The current work only focuses on annotating simple natural language description samples without challenging more complex structured data. For example, the datasets of computational graphs extracted by neural networks consists various nested structure, making it difficult for humans to annotate specific blocks within such complex lists of edges.

Here we discuss a GPT self-supervision approach via a generating-recovering loop, primarily inspired by prompt tuning [20], [21] and the data-centric paradigm [22] The framework of self-supervision method contains the template optimization phase and the summary generation phase. The template optimization stage aims to iteratively find the optimal pairs of the data and summary as the template. Starting with a human-annotated data pair, our iterative process utilizes GPT to generate summaries. The generated summary and raw data form a new data pair. This new data pair is then evaluated for its potential as a template through a comparison of the recovered data from the summary with the original. If both the candidate score (sim^c) and validation score (sim^v) improve, we update the optimal template with the current new data pair. Thus, the self-alignment mechanism iteratively tune the current template for the next round generation. The searched optimal template is subsequently used for generating the summaries on the generation phase. We tune various reward models to evaluate summary quality and introduce various similarity metrics to assess the recovery ability. We perform sufficient experiments on three challenging datasets and conduct detailed ablation study from various perspectives.

Xiaohuan Pei, Yanxi Li, and Chang Xu are with the School of Computer Science, University of Sydney, Camperdown, NSW 2006, Australia. E-mail: xpei8318, yali0722@uni.sydney.edu.au, c.xu@sydney.edu.au.

The results demonstrate our self-supervision paradigm consistently yields competitive performance evaluated by reward models and recovery scores. Additionally, we apply our self-supervision method to generate two new datasets of 3k/15k summaries of the neural network architectures based on the different computational operators.

II. Related Work

Large Language Model. With the breakthrough advancements of Generative Pre-trained Transformer(GPT) family [1]–[3], it is potential to automatically annotate data. The evolution of the Initially, GPT-1 [1] introduced a two-step approach with task-aware input transformations. Subsequently, GPT-2 [2] showcased zero-shot learning without model modifications, and GPT-3 [3] scaled up to enhance performance with minimal fine-tuning. Recently, OpenAI introduced a public tool ChatGPT that utilizes the GPT-3.5 and GPT-4 as engines for generating appropriate responses on various tasks. These models pre-trained by OpenAI have been marked by significant advancements in scale and capabilities [23]–[29]. As revealed by [16]–[18], GPT models can annotate classification tasks through the direct design of diverse prompts. Furthermore, the study by [19] broadens the scope of this annotation task, enabling the generation of dialogues.

Few-Shot Learning. Learning new concepts quickly with limited data is a challenge in machine learning. Few-shot learning constitutes a generalized machine learning paradigm that employs parameter adaptation techniques to optimize model parameters on the basis of a restricted set of supervised examples [30]–[33]. The general paradigm of few-shot learning hinges on its ability to leverage a compact dataset to rapidly train models, often incorporating meta-learning strategies. This approach is distinct in its 'learn to learn' philosophy, aiming to generalize from a few instances rather than relying on extensive data, as seen in conventional machine learning models. Consequently, few-shot learning is particularly relevant in contexts where data availability is limited, offering a strategic advantage in diverse AI applications [30], [34], [35]. As aforementioned, [3] proves that GPT is a powerful few-shot learner, exhibiting remarkable performance on various natural language processing tasks.

In-Context Learning. Similarly, in-context learning paradigm [36]–[38] enables the learning of new concepts from just a few examples without updating the parameters, becoming increasingly applicable alongside the advancing general abilities of large language models. The efficacy of this mode highly relies on the quality and quantity of the example data pairs, which need to be meticulously designed and selected based on the model's comprehension capabilities [36], [39]–[41].

Nonetheless, our approach's application scenario diverges from conventional few-shot and in-context learning scenarios, as the dataset lacks annotated data pairs in the unlabeled candidate set, which shown in Figure 1. Additionally, our work is distinct from prior efforts in contextual learning, which typically rely on initializing the prompt with a set of pre-designed examples, as the example for generation in this work is undefined. The details of difference shown in the Table I. For clarity in our presentation, we adopt the terms 'unlabeled candidate set' and 'validation set' to denote our sample pools. Our sampled data is represented as $\{x, ✓\}$, consisting solely of individual data instances x without the corresponding annotated text y. This approach diverges from the typical data pair format x, y found in the labeled candidate and validation sets.

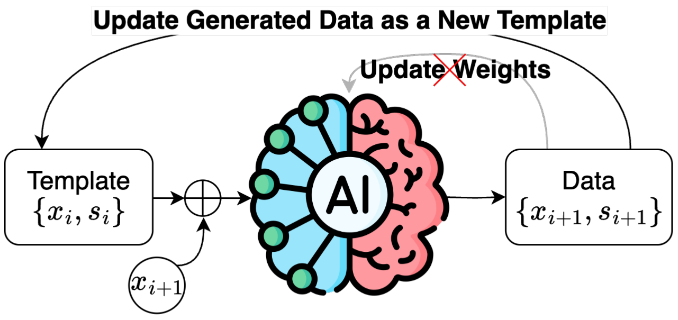

Update Generated Data as a New Template
Update ~~Weights~~
Template $\{x_i, s_i\}$
AI
Data $\{x_{i+1}, s_{i+1}\}$
x_{i+1}

Fig. 1: **Data Centric Generating-Recovering Annotation Paradigm**. Each cycle updates current generated label s_i reused as a part of new template.

TABLE I: Comparison of Scenarios. $\mathcal{X}^{\mathcal{B}}$ denotes support set for tuning in the few-shot learning and $\mathcal{X}^{\mathcal{E}}$ represents the set of designed supervised examples for in-context reference. ΔW: Update Weights. ED: Extensive Data. AD: Annotated Data. SE: Self-Evaluation. **Our work defines a new research paradigm, specifically targeting the third scenario.**

Summary	**w/o ΔW**	**w/o ED**	**w/o AD**	**SE**	**Application Scenarios**
Few Shot	✗	✗	✗	✗	$\{x, y\} \in \mathcal{X}^{\mathcal{B}}$
In Context	✓	✓	✗	✗	$\{x, y\} \in \mathcal{X}^{\mathcal{E}}$
Annotation	✓	✓	✓	✓	$\{x, ?\} \in \mathcal{X}^{\mathcal{E}}, \mathcal{X}^{\mathcal{B}}$

III. Approach

This section delineates our innovative approach to data annotation, which is comprised of two critical stages: the **template optimization stage** and the **summary generation stage**. The former stage focuses on harnessing the self-supervised capabilities of GPT models to derive an optimal template t^* that significantly influences the quality of generated summaries. This is achieved through an iterative process that includes summary generation, data recovery, and feedback comparison to progressively refine the template. The latter stage leverages the refined template to produce concise and accurate summaries for a given dataset. Our methodology aims to balance the precision of data recovery with the fidelity of summarization, resulting in a robust and efficient annotation framework suitable for a variety of domains. The ensuing paragraphs provide a granular view of each stage, detailing the procedures and the underlying algorithms that constitute our approach.

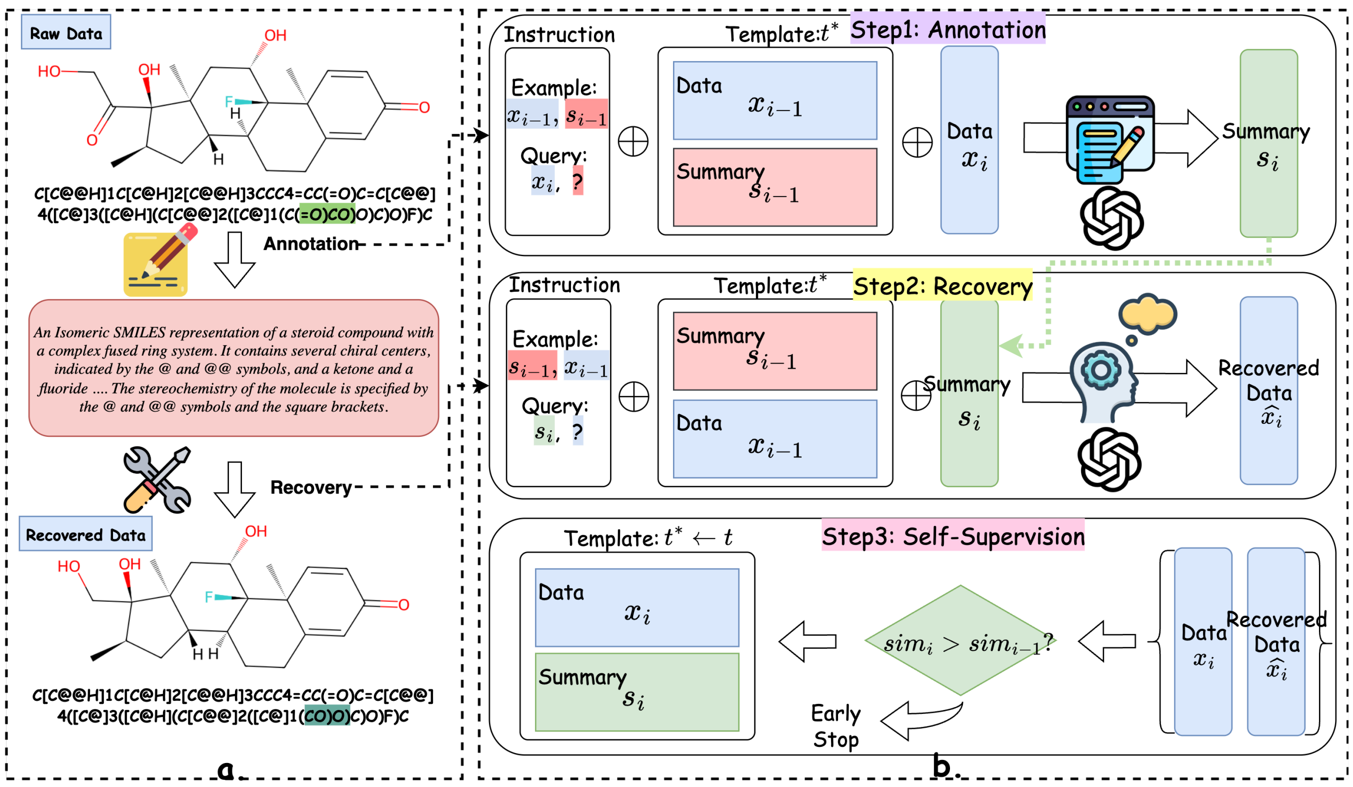

Raw Data
C[C@@H]1C[C@H]2[C@@H]3CCC4=CC(=O)C=C[C@@]4([C@]3([C@H](C[C@@]2([C@]1(C(=O)CO)O)C)O)F)C
Annotation
An Isomeric SMILES representation of a steroid compound with a complex fused ring system. It contains several chiral centers, indicated by the @ and @@ symbols, and a ketone and a fluoride .... The stereochemistry of the molecule is specified by the @ and @@ symbols and the square brackets.
Recovery
Recovered Data
C[C@@H]1C[C@H]2[C@@H]3CCC4=CC(=O)C=C[C@@]4([C@]3([C@H](C[C@@]2([C@]1(CO)O)C)O)F)C
a.
Step1: Annotation
Instruction
Example: x_{i-1}, s_{i-1}
Query: x_i, ?
Template: t^*
Data x_{i-1}
Summary s_{i-1}
Data x_i
Summary s_i
Step2: Recovery
Instruction
Example: s_{i-1}, x_{i-1}
Query: s_i, ?
Template: t^*
Summary s_{i-1}
Data x_{i-1}
Summary s_i
Recovered Data $\widehat{x}_i$
Step3: Self-Supervision
Template: $t^* \leftarrow t$
Data x_i
Summary s_i
$sim_i > sim_{i-1}$?
Early Stop
Data x_i
Recovered Data $\widehat{x}_i$
b.

Fig. 2: An Overview of the ***G**enerating-**R**ecovering **A**nnotation* (**GRA**). **(a.)** illustrates a case of annotating and recovering on the isomeric SMILES. The sample are annotated with natural language summaries and subsequently recovered back to their structural sequence. **(b.)** depicts the framework for self-supervision refinement. Initially, the mechanism concatenates ($\oplus$) the *Instruction*, *Template*, and *Data* into a prompt to annotate with a summary. This generated summary is then used to recover the original data. The template t^* is updated conditionally based on improvements of the recovery capability. If the annotated label is a class, we refine the label s_i format with: $\{$*summary:* $\{\}$, *class:* $\{\}\}$ to enable the model to recover from the description.

A. *Optimize template stage.*

This stage is mainly responsible for finding the optimal template t^* self-supervised by GPT. It is an iterative process of generating a summary, recovering data, and comparing feedback values to navigate the template tuning in the prompt, which is subsequently used for the next round generation.

The iterative process contains the following steps. We first initialize a simple human-labeled pairs of data x_0, summary s_0 as the best template $t^* = \{x_0, s_0\}$, and then respectively assign the role type of $\{$system, assistant, content$\}$ for the instruction, template, candidate data $\{w, t, x\}$. For iteration i, we sample a unlabeled data x_i from the candidate set $\mathcal{X}^c$, which are subsequently concatenated with the current optimal template t^* and default instruction w_g into one message. The GPT $\mathcal{F}(\cdot)$ covert the message to generate summary s_i by referring the instruction, current template and data:

$$s_i \leftarrow \mathcal{F}(x_i|t_i, w_g, \theta), \tag{1}$$

where the θ is the parameters of the language model function and w_g is the instruction of generating a summary. At this point, we obtain a new paired set consisting of data and summary, denoted as $\{x_i, s_i\}$, where s_i is generated from x_i. When the annotated label is a class, the summary s_i could be refined to include both the summary and the label information, enabling the model to recover the data based on the reason for annotating this class.

Considering that a summary's main purpose is to briefly capture the essence of a dataset, the quality of a summary naturally can be deduced from its ability to faithfully reproduce the original dataset. The recovering process is reconstructing $\hat{x}_i$ from s_i by the same GPT:

$$\hat{x}_i \leftarrow \mathcal{F}(s_i|t_i, w_r, \theta), \tag{2}$$

where the θ is the parameters of the large language model and w_g is the instruction of recovering data. And then we apply $sim(\hat{x}_i, x_i)$ to measure the similarity score between the recovered data and original data. If the current score surpasses the previously recorded highest value from iterations, we consider the current data-summary pairs $\{x_i,\ s_i\}$ as a temporary template. Using the same generation and recovery process mentioned in 1 2, we evaluate the average similarity score on the validation set. If this score remains higher than the maximum valid score observed in previous iterations, the self-supervised mechanism update the best template t^* with the current data-summary pairs $\{x_i,\ s_i\}$. Otherwise, the search for the optimal template is terminated early, which could be

Algorithm 1 Generating-Recovering Annotation

```
1: Initialize: instructions $w_g/w_r$ for generating/recovering , initialized a template $t^* \leftarrow \{x_0, s_0\}$, $sim^c \leftarrow 0$, $sim^v \leftarrow 0$, a candidate unlabeled set $\mathcal{X}^c$ for sampling, a validation set $\mathcal{X}^v$ for testing the current template performance.
2: for iteration $i \leftarrow 1$ to $I$ do
3:     $x_i \leftarrow \mathcal{X}^c$                          ▷ Sampling an un-labeled data
4:     $s_i \leftarrow GPT(\langle w_g, t^*, x_i \rangle)$       ▷ Annotating $x_i$ by (Eq. 1)
5:     $\hat{x}_i \leftarrow GPT(\langle w_r, t^*, s_i \rangle)$   ▷ Recovering $x_i$ by (Eq. 2)
6:     $sim_i^c \leftarrow sim(x_i, \hat{x}_i)$          ▷ Alignment by $sim(\hat{x}_i, x_i)$
7:     if $sim_i^c > sim^c$ then
8:         Compute $sim_i^v$  ▷ Repeat annotating-recovering process on $\mathcal{X}^v$
9:         if $sim_i^v > sim^v$ then
10:            $sim^c \leftarrow sim_i^c$
11:            $sim^v \leftarrow sim_i^v$
12:            $t^* \leftarrow \{x_i, s_i\}$      ▷ Update template $t^*$ (Fig. 2 (b.))
13:        end if
14:    end if
15: end for
```

denoted as:

$$t_i^* = \begin{cases} \{x_i, s_i\}, & \text{If } sim(\hat{x}_i, x_i) > sim(\hat{x}_{i-1}, x_{i-1}) \\ t_{i-1}^*, & \text{Otherwise,} \end{cases} \tag{3}$$

where t_{i-1}^* is the best candidate template in the last round. The process continues until the current template cannot generate a higher recovery score compared to the previous iteration or when the maximum number of iterations is achieved. The self-supervised objective of above iterative updating is to find the best template t_i that maximizes the expected similarity between the recovered data and the original data. The optimization process can be formalized as follows:

$$t^* = \arg\max_t \ \mathbb{E}_{x_i \sim \mathcal{X}^c}[sim(\hat{x}_i, x_i)], \tag{4}$$

where the $sim(\cdot)$ is the metric of similarity function between two sequence data. This objective assumes that the similarity $sim(\cdot)$ is a meaningful measure of the quality of the recovery process, and that higher values of similarity correspond to better recoveries. Unlike conventional one-shot learning where model weights are updated during training, our annotation approach updates the current template instead, with newly annotated data pairs.

B. Summary generation stage.

In this stage, GPT concatenates the identified optimal template with instructions, using it as the optimal prompt to generate natural language summaries for the generation dataset:

$$s \leftarrow \mathcal{F}(x_i | t^*, w_g), \tag{5}$$

where s is the annotated summary of x_i. We test the summary quality by tuning various human feedback reward networks, and then we introduce the recovery evaluation aiming to measure whether the summary could decoding the high-level sentences to original data. Specifically, we assume that the stronger the ability to recover the intermediate summary back to the original data, the more it substantiates the professionalism and accuracy of the summarises. Conversely, a weaker recovery capability implies a lower degree of professionalism and accuracy in the summarises.

The evaluation phase of this process is two-fold. Firstly, we assess the quality of the summary by employing a variety of human feedback reward networks which offers us a comprehensive insight into the real-world applicability and understandability of our summaries. Following this, we institute a recovery evaluation process that seeks to measure the efficiency and accuracy with which the summary can decode high-level sentences back into the original data, thereby serving as an indicator of the professionalism and accuracy of the summarises. This assumption underpins our belief that the more capable the model is of reverse engineering the summary back to the original data, the more it affirms the precision and professionalism of the summaries. Conversely, if the model demonstrates a lower proficiency in this restoration task, it suggests that the summaries lack a certain degree of professionalism and accuracy. The evaluation phase of our approach is two-part. Initially, we utilize various human feedback reward networks to assess the quality of the summaries, providing us with insights into how well they might perform in practical settings and how understandable they are to potential users. This step examines the real-world utility and user-friendliness of our summaries. Next, we introduce a recovery evaluation to quantitatively test how accurately the summaries can be translated back into the original data. This step acts as a measure of the consistency of the summaries.

C. Convergence Analysis

Our Generating-Recovering Annotation (GRA) framework aims to iteratively optimize a template t^* to maximize the similarity between the original data x and the recovered data $\hat{x}$. Given a dataset of candidate unlabeled data $\mathcal{X}^c$, a validation set $\mathcal{X}^v$, and a similarity function $\text{sim}(x, \hat{x}) \in [0, 1]$, the objective is defined as:

$$t^* = \arg\max_t \mathbb{E}_{x \in \mathcal{X}^v} \left[\text{sim}(x, \hat{x})\right]. \tag{6}$$

At each iteration i, a data sample x_i is drawn from $\mathcal{X}^c$, and the sample is annotated to produce s_i using the current template t^*, following $s_i \sim p(s \mid x_i, t^*)$. The resulting annotation s_i is then used to recover the data $\hat{x}_i$, modeled as $\hat{x}_i \sim p(\hat{x} \mid s_i, t^*)$. Next, the similarity between the original data and the recovered data is computed as $\text{sim}_i^c = \mathbb{E}_{s \sim p(s|x_i, t^*)} \mathbb{E}_{\hat{x} \sim p(\hat{x}|s, t^*)} \left[f(x_i, \hat{x})\right]$, where $f(x, \hat{x})$ quantifies alignment, such as reconstruction fidelity or semantic similarity. Finally, the validation similarity is calculated using $\text{sim}_i^v = \frac{1}{|\mathcal{X}^v|} \sum_{x \in \mathcal{X}^v} \text{sim}(x, \hat{x})$, where $\hat{x}$ is generated based on the updated t^*.

The template t^* is updated only if the validation similarity improves:

$$t^* \leftarrow \{x_i, s_i\}, \quad \text{if } \text{sim}_i^v > \text{sim}^v. \tag{7}$$

The similarity sequence $\{\text{sim}_i^v\}_{i=1}^{\infty}$ is bounded in $[0, 1]$ and monotonic ($\text{sim}_{i+1}^v \geq \text{sim}_i^v$), ensuring convergence:

$$\lim_{i \to \infty} \text{sim}_i^v = \text{sim}^*, \tag{8}$$

where sim* is the maximum similarity achievable under the framework. So the optimization objective can be reformulated as:

$$t^* = \arg\max_t \mathbb{E}_{x\in\mathcal{X}^v} \int_s p(s \mid x,t) \int_{\hat{x}} p(\hat{x} \mid s,t) f(x,\hat{x})\, d\hat{x}\, ds. \tag{9}$$

At each iteration, the improvement in validation similarity can be bounded as:

$$\mathrm{sim}_{i+1}^v - \mathrm{sim}_i^v \geq \mathbb{E}_{x\in\mathcal{X}^v} \left[\int_s \Delta p(s \mid x,t) \int_{\hat{x}} p(\hat{x} \mid s,t) f(x,\hat{x})\, d\hat{x}\, ds\right], \tag{10}$$

where $\Delta p(s \mid x,t)$ reflects the change in annotation probability induced by the template update. As the improvement $\mathrm{sim}_{i+1}^v - \mathrm{sim}_i^v$ diminishes with each iteration, the template updates $\Delta t_i = t^{(i+1)} - t^{(i)}$ converge:

$$\lim_{i\to\infty} \Delta t_i = 0. \tag{11}$$

Thus, the GRA framework guarantees convergence to an optimal template t^*, ensuring maximum alignment between the original and recovered data over the validation set. This iterative process leverages the monotonicity of similarity improvements and the boundedness of $\mathrm{sim}(x,\hat{x})$ to achieve stability and convergence.

IV. Experiments

A. Experiment Setup

Dataset. This study utilizes three distinct unlabeled datasets: Darts-Medium, Darts-Large, and PubMed. The Darts-Medium and Darts-Large datasets consist of neural architecture networks, generated by 5 and 7 operators respectively, with Darts-Large having more nodes. They provide a rich source of information on the design and performance of various neural architectures. We have also incorporated the PubMed dataset which consists of Isomeric SMILES structures into our study. This dataset represents a different domain, offering an opportunity to examine the generation of high-level summaries for complex chemical structures. We take great care to avoid test set leakage and split data for each stage to use. Each of these datasets has been divided into two segments for different stages of our experiment: the template optimization phase and the generation phase. The template optimization stage involves a candidate set and a validation set, each containing 50 samples. We also conduct class label annotation on the widely used benchmark SemEval [42]. The results and analysis are shown in the Appendix of the supplementary material.

TABLE II: Summary of Annotated Datasets

Dataset	Template Tuning Stage		Generation Stage	Sequence Element
	Candidate	Valid	Annotation	
Darts-Large	50	50	12,362	7 Operators
Darts-Medium	50	50	2,972	5 Operators
PubMed	50	50	4,111	Isomeric SMILES

Setup. We implement a temperature hyperparameter of 1 to promote diversity in the summary generation when evaluating baseline performance. Conversely, during the data generation stage, the temperature was set to zero to ensure consistency and stability. To test the performance of different large language models, we apply four widely used models in this experiment, composed of the version 3 of GPT: davinci, text-curie-001 and version 3.5 of GPT: text-davinci-003, gpt-3.5-turbo. We call the response through the inference of the openAI official APIs [1]. The initialized information is divided into instructions, templates and query data, respectively calling the role of system, assistant and user in the information flow. We set 10 iterations in each template tuning phase with a defined maximum token length of 350 for generating summaries and 500 for recovering datas from the summaries. Each input prompt consists of three parts, an instruction, a template and a sampled data. We give the input and output formats in the instruction, and define the meaning of each structured symbol in the data. This part costs 500 tokens. For the template, we assign assistant and content tags to the data and summary in the template, so that GPT can recognize that this is a template information, and this part costs 3000 tokens. For sampling data from unlabeled candidate set in the template optimization phase, we directly use the role of user to send it to GPT. In order to make the generated summary more diverse in the template tuning stage, we set the hyperparameter of temperature to 1. In the data generation stage, we set the temperature to 0 to keep it stable.

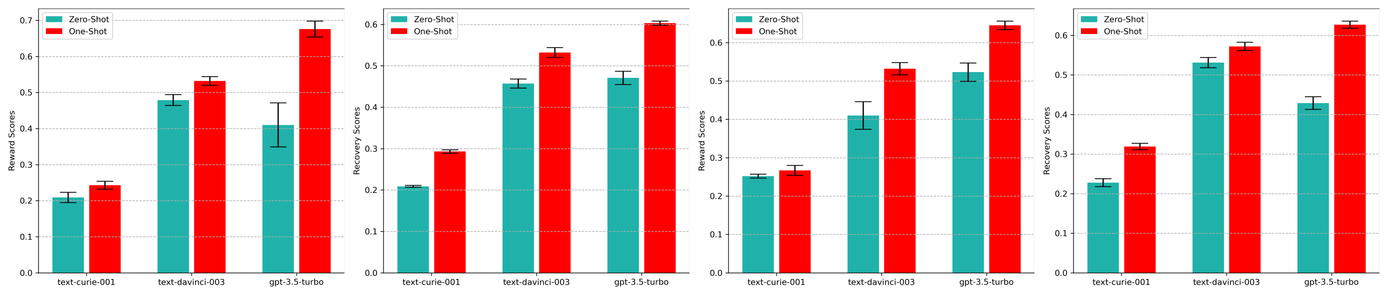
TABLE III: Generated Summary Evaluation scores (± standard error) on the three datasets.

Model	Generated Summary Evaluation			
	R1	R2	R3	R4
Darts-Medium				
davinci	$0.296_{\pm 0.012}$	$0.302_{\pm 0.041}$	$0.194_{\pm 0.016}$	$0.288_{\pm 0.019}$
text-curie	$0.243_{\pm 0.028}$	$0.211_{\pm 0.013}$	$0.297_{\pm 0.004}$	$0.342_{\pm 0.013}$
text-davinci	$0.532_{\pm 0.023}$	$0.596_{\pm 0.028}$	$0.503_{\pm 0.032}$	$0.582_{\pm 0.031}$
gpt-3.5-turbo	$\mathbf{0.513}_{\pm 0.011}$	$\mathbf{0.642}_{\pm 0.016}$	$\mathbf{0.519}_{\pm 0.015}$	$\mathbf{0.639}_{\pm 0.017}$
Darts-Large				
davinci	$0.302_{\pm 0.024}$	$0.294_{\pm 0.053}$	$0.197_{\pm 0.023}$	$0.305_{\pm 0.025}$
text-curie	$0.252_{\pm 0.038}$	$0.220_{\pm 0.023}$	$0.305_{\pm 0.011}$	$0.370_{\pm 0.021}$
text-davinci	$0.509_{\pm 0.034}$	$0.607_{\pm 0.021}$	$0.515_{\pm 0.011}$	$0.580_{\pm 0.027}$
gpt-3.5-turbo	$\mathbf{0.544}_{\pm 0.020}$	$\mathbf{0.672}_{\pm 0.037}$	$\mathbf{0.537}_{\pm 0.021}$	$\mathbf{0.657}_{\pm 0.033}$
PubMed				
davinci	$0.318_{\pm 0.017}$	$0.354_{\pm 0.026}$	$0.216_{\pm 0.032}$	$0.310_{\pm 0.031}$
text-curie	$0.262_{\pm 0.013}$	$0.230_{\pm 0.021}$	$0.327_{\pm 0.018}$	$0.372_{\pm 0.026}$
text-davinci	$\mathbf{0.547}_{\pm 0.019}$	$0.611_{\pm 0.022}$	$0.514_{\pm 0.036}$	$0.594_{\pm 0.023}$
gpt-3.5-turbo	$0.542_{\pm 0.022}$	$\mathbf{0.671}_{\pm 0.031}$	$\mathbf{0.547}_{\pm 0.023}$	$\mathbf{0.667}_{\pm 0.012}$

B. Evaluation

Summary Evaluation. To directly evaluate summary quality generated by our approach, we tuned various human feedback reward models as our evaluators. These reward models

[1]platform.openai.com/docs/models/



(a) Reward Scores on Darts (b) Recovery Scores on Darts (c) Reward Scores on PubMed. (d) Recovery Scores on PubMed.

Fig. 3: The role of optimizing template navigation: a comparative analysis of GAR vs. zero-shot approaches. The red bars demonstrate initial the prompt with adding a template, and the green bars represent the scores conditional on the same instruction without a template. We use GPT-3.5-turbo to conduct the ablation study.

have been widely acknowledged in the literature for their effectiveness in providing evaluative feedback for language generation tasks. Specifically, we tuned various human feedback reward models, as our evaluators, which is detailed in the Appendix. Each evaluator evaluates the quality of the summary and provides a corresponding reward score ($R1, R2, R3, R4$), and the details of the reward process are consistent with the original work [43]. To ensure stability in the reward distribution, we employed a k-fold cross-validation strategy with a value of k set to 5. Assume $r(x, s|\theta)$ represents the scalar output of the reward model for data x and summary s, parameterized by θ. We performed fine-tuning on each pretrained reward network by following the steps outlined in [43]:

$$\mathcal{L} = \mathbb{E}_{(x,s_0,s_1,s_i)\sim\mathcal{D}}[log(\sigma(r(x,s_i|\theta)) - \sigma(r(x,s_{1-i}|\theta))], \quad (12)$$

where $i \in \{0, 1\}$ and $\mathcal{D}$ represent the human-labelled datasets containing judgments on which summary generated by two large language models is superior. The equation delineated above illustrates the loss function that we used during this fine-tuning process.

Recovery Evaluation. We implement both sentence-level alignments and embedding-level metrics to assess the discrepancy between the recovered data and the original data. For the sentence-level evaluation, we employed the average BLEU score [44] and ROUGE-L [45], [46]. Specifically, we utilize a smoothed average version of BLEU in our evaluations to counteract the issues that can arise with BLEU when dealing with short sentences. ROUGE-L, on the other hand, is based on Longest Common Subsequence (LCS) statistics, which makes it a robust tool for evaluating the quality of summaries, particularly in our case where it was applied for the evaluations. In terms of embedding-level metrics, we employ Semantic Textual Similarity (STS) [47] embedding and Bidirectional Encoder Representations from BERT [48] embedding. In particular, STS embedding offers a quantifiable measure of the semantic equivalence between two textual sequences, which is ideal for assessing the semantic similarity between the original and recovered data. On the other hand, the BERT embedding pre-trained from the BERT model, allows us to capture more nuanced semantic and syntactic features of the data structures, providing a more comprehensive and insightful analysis of our generated summaries compared to the original data.

TABLE IV: Recovered Data Evaluation scores (± standard error) on the three datasets.

Model	Recovered Data Evaluation			
	BLEU	ROUGE	STS Sim	Bert Sim
Darts-Medium				
davinci	$0.195_{\pm 0.004}$	$0.214_{\pm 0.007}$	$0.291_{\pm 0.001}$	$0.263_{\pm 0.005}$
text-curie	$0.118_{\pm 0.004}$	$0.172_{\pm 0.008}$	$0.310_{\pm 0.006}$	$0.294_{\pm 0.009}$
text-davinci	$0.278_{\pm 0.006}$	$0.379_{\pm 0.017}$	$0.772_{\pm 0.003}$	$0.543_{\pm 0.025}$
gpt-3.5-turbo	$\mathbf{0.482}_{\pm 0.023}$	$\mathbf{0.422}_{\pm 0.004}$	$\mathbf{0.816}_{\pm 0.014}$	$\mathbf{0.691}_{\pm 0.002}$
Darts-Large				
davinci	$0.194_{\pm 0.012}$	$0.221_{\pm 0.017}$	$0.287_{\pm 0.009}$	$0.268_{\pm 0.045}$
text-curie	$0.129_{\pm 0.014}$	$0.182_{\pm 0.022}$	$0.314_{\pm 0.013}$	$0.308_{\pm 0.019}$
text-davinci	$0.292_{\pm 0.005}$	$0.382_{\pm 0.013}$	$0.781_{\pm 0.001}$	$0.559_{\pm 0.009}$
gpt-3.5-turbo	$\mathbf{0.505}_{\pm 0.030}$	$\mathbf{0.447}_{\pm 0.009}$	$\mathbf{0.829}_{\pm 0.019}$	$\mathbf{0.715}_{\pm 0.006}$
PubMed				
davinci	$0.209_{\pm 0.015}$	$0.228_{\pm 0.002}$	$0.293_{\pm 0.002}$	$0.265_{\pm 0.002}$
text-curie	$0.138_{\pm 0.012}$	$0.192_{\pm 0.001}$	$0.316_{\pm 0.001}$	$0.310_{\pm 0.001}$
text-davinci	$\mathbf{0.571}_{\pm 0.008}$	$0.403_{\pm 0.001}$	$0.774_{\pm 0.004}$	$0.571_{\pm 0.002}$
gpt-3.5-turbo	$0.509_{\pm 0.004}$	$\mathbf{0.457}_{\pm 0.001}$	$\mathbf{0.796}_{\pm 0.001}$	$\mathbf{0.635}_{\pm 0.002}$

C. Baseline Results

We conducted experiments on three highly challenging datasets using four different models: davinci, text-curie-001, text-davinci-003, gpt-3.5-turbo. The evaluation included testing the summary quality by four specific reward networks and assessing the data recovery capability.

Evaluate on the Recovery Data. We first focus on the evaluation of the generated summary using the four reward scores. These scores represent the performance of the generated summary quality by various human feedback reward models after fine-tuning. On the Darts-Medium dataset, GPT-3.5-Turbo outperforms the other models with the highest scores across all metrics: a BLEU score of 0.482 ± 0.023, a ROUGE score of 0.422 ± 0.004, a STS Sim score of 0.816 ± 0.014, and a Bert Sim score of 0.691 ± 0.002. The Text-davinci-003 model followed next, while the Davinci and Text-curie-001 models lag behind, with lower scores on all measures. Similar trends were observed on the Darts-Large dataset. Once again, GPT-3.5-Turbo displays superior performance, achieving the highest scores in all categories: BLEU (0.505 ± 0.030), ROUGE (0.447 ± 0.009), STS Sim (0.829 ± 0.019), and Bert Sim (0.715 ± 0.006). Text-davinci-003 maintains its second place ranking, while Davinci and Text-curie-001 trails with less impressive scores. On the PubMed dataset, the Text-davinci-003 model remarkably achieved the highest BLEU score of 0.571 ± 0.008, surpassing the other models. However, GPT-3.5-Turbo still led the other metrics, with a ROUGE score of 0.457 ± 0.001, a STS Sim score of 0.796 ± 0.001, and a Bert Sim score of 0.635 ± 0.002. Davinci and Text-curie-001 continued to exhibit inferior performance compared to the other models. While the performance varied somewhat across different datasets, GPT-3.5-Turbo consistently yields the strongest results on the evaluated metrics. This strongly indicates its superior capability in data recovery tasks. The performance gap observed between the models highlights the importance of selecting the right transformer model for specific tasks and datasets, thereby optimizing the trade-off between computational resources and performance.

Evaluate on the Summary Quality. The results in Table III demonstrate that on the Darts-Medium dataset, Text-davinci-003 displays the best overall performance in terms of R1 (0.532 ± 0.023), R2 (0.596 ± 0.028), and R3 (0.503 ± 0.032) scores. However, GPT-3.5-Turbo achieves the highest R4 score (0.639±0.017). The other models, Davinci and Text-curie-001, have lower scores across these metrics. On the Darts-Large dataset, GPT-3.5-Turbo outperformed the other models across all reward metrics, with R1 (0.544±0.020), R2 (0.672±0.037), R3 (0.537 ± 0.021), and R4 (0.657 ± 0.033) scores. Text-davinci-003 follows closely behind, while Davinci and Text-curie-001 lag further in terms of performance. For the PubMed dataset, Text-davinci-003 demonstrate superior performance in the R1 (0.547 ± 0.019), R2 (0.611 ± 0.022), and R3 (0.514 ± 0.036) metrics, while GPT-3.5-Turbo achieved the highest R4 score (0.667 ± 0.012). As in the previous datasets, Davinci and Text-curie-001 exhibit lower scores across these reward metrics. Based on the performance, the evaluation on the reward models reveals that Text-davinci-003 and GPT-3.5-Turbo consistently outperform the other models in terms of R1, R2, R3, and R4 scores. These findings emphasize the importance of selecting the appropriate reward models for specific tasks and datasets, as they have a significant impact on the quality of the generated summaries.

V. Ablation Study

In the ablation study, we primarily aim to address four key questions for the proposed annotation approach:

Q1. What role does the one-shot template play, and can the same effects be achieved via zero-shot method or generating summaries only with designed instruction? **Q2.** If the template work in the self-supervised annotation, what impact do various similarity measurement methods have on the outcomes of one-shot tuning? **Q3.** How does the initialization of the template affect the results of the optimal one in the one-shot tuning phase? **Q4.** Are the generated summary influenced by the hyperparameters of the GPT model itself?

TABLE V: The impact of the measurement metrics.

	STS	**BERT**	**ROUGE**	**BLEU**	**Darts**	**PubMed**
	✓				52.14	58.89
			✓		53.21	60.32
$\overline{R}$	✓		✓		61.48	60.05
	✓	✓	✓		**62.59**	65.27
	✓	✓	✓	✓	61.17	**65.96**
	✓				21.18	19.04
			✓		26.19	25.20
$\overline{S}$	✓		✓		29.44	39.55
	✓	✓	✓		**45.59**	40.04
	✓	✓	✓	✓	44.05	**41.38**

A. The impact of the optimal template.

Our study systematically contrasts the zero-shot and GAR strategies across diverse models, while also focusing on two domain-specific datasets to ensure a thorough investigation. The evaluative framework we adopted adheres to the procedural blueprint delineated in Section IV-B. In Figure 3, we present an exhaustive comparative analysis between these two generative methodologies, taking into account both the intrinsic quality of the directly generated annotations and the accuracy of the subsequent data recovery process. The records clearly indicate that the GAR approach (indicated in red within the figure) benefits from iterative template refinement, and markedly outperforms its zero-shot counterpart (depicted in blue) with respect to the quality of data annotation. This superiority holds true whether we assess the performance based on the reward model's outcomes or through the fidelity of the data reconstructed from the annotations. Furthermore, by implementing cross-validation within the data generation phase and observing the standard deviation across various dataset splits, we have identified a pattern of smaller red error bars relative to the blue ones within the figure. The empirical evidence thus supports the assertion that incorporating an optimally tuned the template into the annotation process significantly boosts performance metrics when compared to generating annotations in a zero-shot manner.

B. The influence of the measurement metrics

The feedback process of our proposed iterative algorithm is based on the measure of similarity between the recovered data and the original data, as to the choice of similarity metrics is essential. Considering that both the structured original data and the generated data are sequence information, we apply the most widely used schemes for measuring sequence similarity, ranging from sentence-level measurement methods to embedding-level measurement techniques. The experimental pipeline for this ablation study adheres to the previous process. Table V provides the records of the testing scores by setting different similarity scores as the feedback. Initially, we tested the effects of two different types of single similarity calculation functions on the annotated results. The first two rows of each group suggest that sentence structure has a slightly more

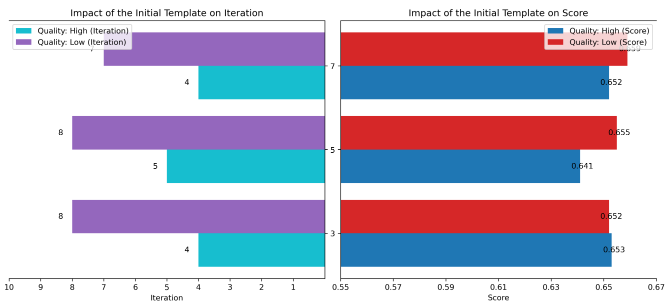


Fig. 4: Impact of the initial template. From the view of the complexity of the initial data and summary.

positive impact on the results. Moreover, by comparing the last two rows (mixed metric functions) with the first two rows (single metric function) in each group, we observe that whether we directly measure with the reward function or indirectly assess the summary's recovery ability, enhancing the diversity of metric functions can effectively improve the quality of the annotated data.

C. *The quality of the initial templates*

Our empirical observations have led us to conclude that the initial conditions of the templates play a crucial role in influencing the training dynamics. To investigate this phenomenon in depth, we conducted a comprehensive experiment on the Darts dataset. In this experiment, we deliberately crafted templates with varying complexity levels, specifically focusing on templates formed using 3, 5, and 7 operators. Further, we classified these templates into two distinct categories based on summary quality: High-Quality Annotation and Low-Quality Annotation. This classification resulted in a total of six unique experimental conditions to thoroughly assess the impact of initial template quality. For each condition, we run the template 50 times in a controlled environment, ensuring consistency across trials. To guarantee stable and consistent summary generation, we fixed the GPT temperature parameter at 0. This setup allowed us to isolate the effects of initial template quality on the learning process, particularly examining how high-quality initial templates contribute to faster convergence while potentially leading to lower variance in similarity scores in the final iterations of the training. As shown in Figure 4, varying initial conditions significantly influence the training dynamics. Warm colored bars indicate that there are fewer iterations needed for high-quality summary generation compared to low-quality templates. However, the final similarity scores are similar regardless of the quality or complexity of the initial template. This suggests two conclusions: starting with a high-quality template accelerates convergence, enhancing algorithm efficiency; and final template quality is not heavily dependent on the initial template, indicating initial conditions primarily affect convergence rate rather than final outcome quality.

D. *The impact of the temperature*

In order to comprehensively investigate the impact of temperature hyperparameters on the performance of different models, we conducted a series of experiments focusing on the convergence properties of four baseline models under varying temperature settings in different environments. To ensure the robustness of our findings, we performed each run 30 times, tracing the standard deviation of the scores. Figure 5 reveal a consistent trend across all models: when the temperature is set to 0, the iterative search process terminates prematurely at the third or fourth iteration. This suggests an inclination towards a limited exploration space, leading to the generation of less diverse outputs. On the other hand, elevating the temperature value to 1 during the template optimization stage demonstrated an interesting outcome. Although each iteration's pace was reduced, this facilitated a broader array of alternatives for subsequent template generation. This indicates an expansion of the exploration space, allowing for the generation of more diverse and potentially creative solutions. We also observed that setting the temperature to 1 resulted in higher final scores compared to a setting of 0. This finding suggests that optimal templates can be obtained by properly tuning the temperature hyperparameter to control the diversity of the generated summaries.

TABLE VI: Performance comparison of GPT-Turbo-3.5 with and without the GAR paradigm on the SemEval 2017 dataset

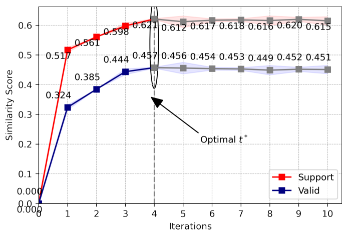
Method	Accuracy
GPT-Turbo-3.5-1106	0.9732
GPT-Turbo-3.5-1106 (+GAR)	0.9803

E. *Classification Results*

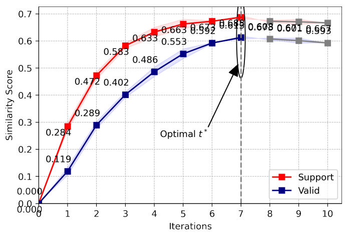
Our contribution primarily focuses on professional academic sequence annotation with a summary. However, it can also be applied to other classification annotation tasks with some modifications to the instructions. For instance, in the context of a classification task, the scenario changes: the inputs become a subject textual sentence. We consider that the output—a label—may be hard to recover. However, we can enhance the recovery process by adding a reasoning description of the question. For example, we could modify the recovery instructions to a format design of return {*description:* {}, *label:* {}}. We follow the settings of the sentimentGPT [49] and apply GAR on the widely used dataset SemEval 2017 [42]. The model utilized GAR paradigm to find an optimal template and achieve better scores compared to directly annotating data. The instruction design of the classification task on the SemEval demonstrated as:

- **[Generating]** *Analyze the following product review and determine if the sentiment is: positive or negative. Return answer is the description of query sentence and a single word as either positive or negative:* {*description*} {*text*}
- **[Recovering]** *Recover the sentence without the label of positive or negtive, given by an answer of sentimental analysis.*

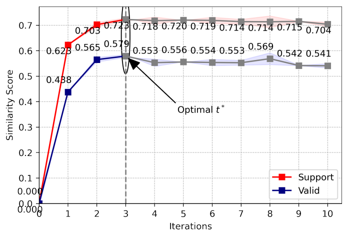
Table VI shows that integrating the GAR paradigm improves GPT-Turbo-3.5's accuracy from 0.9732 to 0.9803 on the SemEval 2017 dataset, highlighting the effectiveness of reasoning-based instructions.



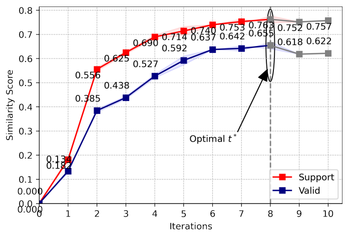
(a) T=0 of text-curie-001 on Darts-Medium;



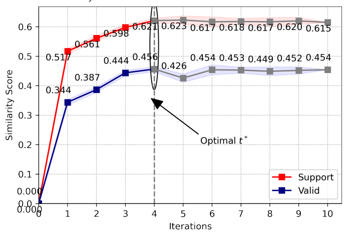
(b) T=1 of text-curie-001 on Darts-Medium;



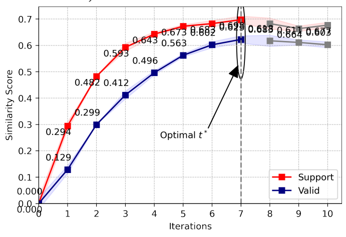
(c) T=0 of gpt-3.5-turbo on the Darts-Medium;



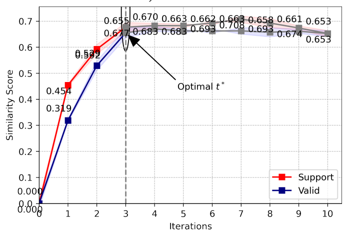
(d) T=1 of gpt-3.5-turbo on the Darts-Medium;



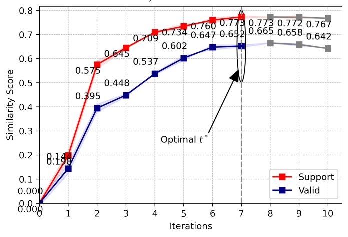
(e) T=0 of davinci on the Darts-Medium;



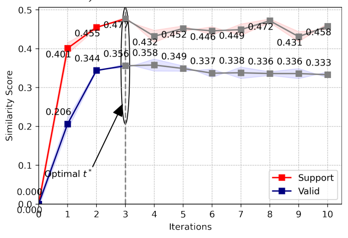
(f) T=1 of davinci on the Darts-Medium;



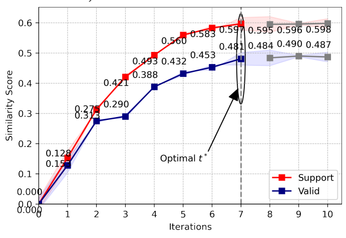
(g) T=0 of text-davinci-003 on the Darts-Medium;



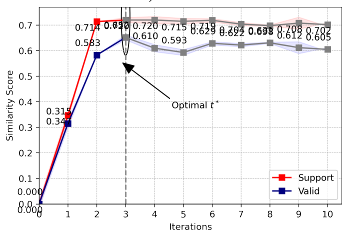
(h) T=1 of text-davinci-003 on the Darts-Medium;



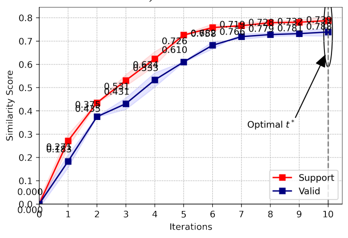
(i) T=0 of text-curie-001 on the PubMed;



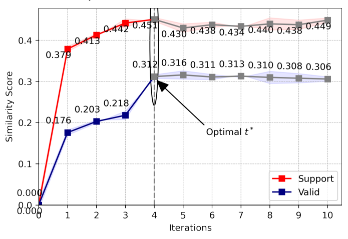
(j) T=1 of text-curie-001 on the PubMed;



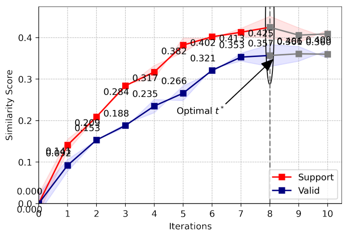
(k) T=0 of gpt-3.5-turbo on the PubMed;



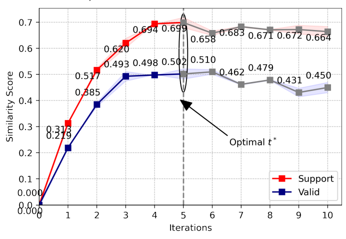
(l) T=1 of gpt-3.5-turbo on the PubMed;



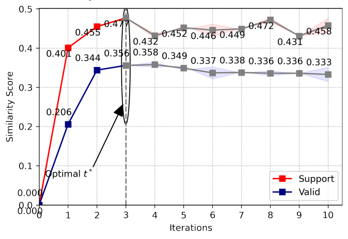
(m) T=0 of davinci on the PubMed;



(n) T=1 of davinci on the PubMed;



(o) T=0 of text-davinci-003 on the PubMed;



(p) T=1 of text-davinci-003 on the PubMed;

Fig. 5: The Impact of the model's temperature. The red lines represent the iterative records of similarity score by current candidate data and the blue lines trace the recovery scores on the validation set. The arrow points the average number of the iteration to find the optimal template.

VI. Annotation Cases

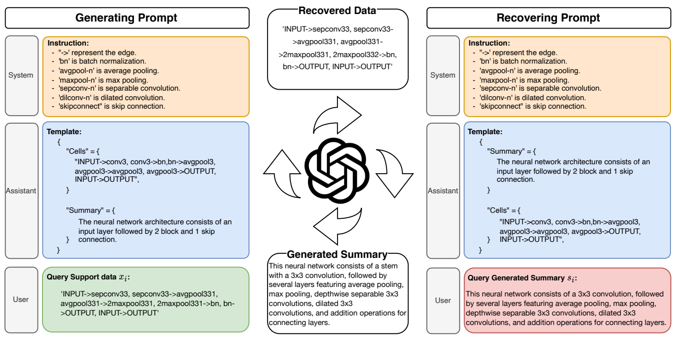
Here we provide two cases to elaborate how GPT self-supervised annotating the complex structured data on the Darts and PubMed dataset. We first demonstrate the prompt for generating summary and recovering data, and then we demonstrate the pipelines of our approach.

a) Prompts: The generating prompt comprises three elements: encoding instruction, template, query data, while the recovering prompt contains decoding instruction, template, query summary.

b) Pipeline: We use the case 6 on the Darts dataset to elaborate on the pipeline of our annotation approach. *Step 1.* Initialize a template composed of cells and a summary. *Step 2.* Sample a set of cells from the support set. *Step 3.* Concatenate the <*encoding instruction, template, cells*> into one message. *Step 4.* Send the message to GPT and receive a summary response. *Step 5.* Concatenate the <*decoding instruction, template, summary*> into one message. *Step 6.* Send the message to GPT and receive a response of recovered cells. *Step 7.* Compute the similarity between the recovered cells and original cells. If it is larger than the previous record, update the score and treat the sampled data and generated summary as a temporary template. *Step 8.* Evaluate the temporary template on the valid set by repeating the above process. *Step 9.* If the average scores on the validation dataset also surpass previous records, update the best valid score and replace the current template with the temporary template. *Step 10.* Iteratively repeat *Step 2.* to *Step 9.* until the maximum number of iterations is reached.

VII. Conclusion

This paper introduces a new approach, GPT self-supervision annotation (GAR), which harnesses the few shot learning



Instruction:
- "->' represent the edge.
- 'bn' is batch normalization.
- 'avgpool-n' is average pooling.
- 'maxpool-n' is max pooling.
- 'sepconv-n' is separable convolution.
- 'dilconv-n' is dilated convolution.
- 'skipconnect" is skip connection.

```
Template:
  {
    "Cells" = {
       "INPUT->conv3, conv3->bn,bn->avgpool3,
       avgpool3->avgpool3, avgpool3->OUTPUT,
       INPUT->OUTPUT",
    }

    "Summary" = {
        The neural network architecture consists of an
        input layer followed by 2 block and 1 skip
    }   connection.
  }
```

Query Support data x_i:

'INPUT->sepconv33, sepconv33->avgpool331, avgpool331->2maxpool331, 2maxpool331->bn, bn->OUTPUT, INPUT->OUTPUT'

Recovered Data: 'INPUT->sepconv33, sepconv33->avgpool331, avgpool331->2maxpool331, 2maxpool332->bn, bn->OUTPUT, INPUT->OUTPUT'

Generated Summary: This neural network consists of a stem with a 3x3 convolution, followed by several layers featuring average pooling, max pooling, depthwise separable 3x3 convolutions, dilated 3x3 convolutions, and addition operations for connecting layers.

Recovering Prompt

Instruction:
- "->' represent the edge.
- 'bn' is batch normalization.
- 'avgpool-n' is average pooling.
- 'maxpool-n' is max pooling.
- 'sepconv-n' is separable convolution.
- 'dilconv-n' is dilated convolution.
- 'skipconnect" is skip connection.

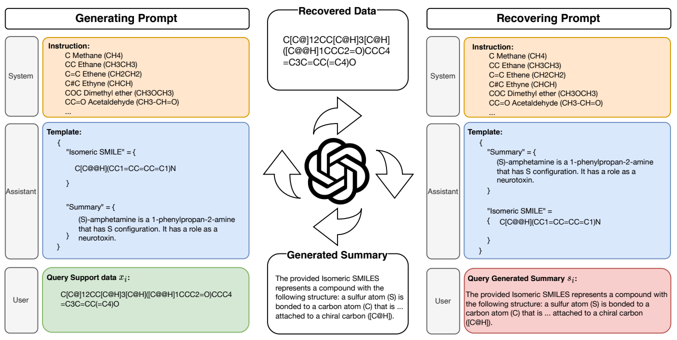
```
Template:
  {
    "Summary" = {
        The neural network architecture consists of an
        input layer followed by 2 block and 1 skip
        connection.
    }

    "Cells" = {
        "INPUT->conv3, conv3->bn,bn->avgpool3,
        avgpool3->avgpool3, avgpool3->OUTPUT,
    }   INPUT->OUTPUT",
  }
```

Query Generated Summary s_i:

This neural network consists of a 3x3 convolution, followed by several layers featuring average pooling, max pooling, depthwise separable 3x3 convolutions, dilated 3x3 convolutions, and addition operations for connecting layers.

Fig. 6: Darts datasets: A case of the generating-recovering annotation.



```
Instruction:
    C Methane (CH4)
    CC Ethane (CH3CH3)
    C=C Ethene (CH2CH2)
    C#C Ethyne (CHCH)
    COC Dimethyl ether (CH3OCH3)
    CC=O Acetaldehyde (CH3-CH=O)
    ...
```

```
Template:
  {
    "Isomeric SMILE" = {

       C[C@@H](CC1=CC=CC=C1)N

    }

    "Summary" = {
        (S)-amphetamine is a 1-phenylpropan-2-amine
        that has S configuration. It has a role as a
    }   neurotoxin.
  }
```

Query Support data x_i:

C[C@]12CC[C@H]3[C@H]([C@@H]1CCC2=O)CCC4=C3C=CC(=C4)O

Recovered Data: C[C@]12CC[C@H]3[C@H]([C@@H]1CCC2=O)CCC4=C3C=CC(=C4)O

Generated Summary: The provided Isomeric SMILES represents a compound with the following structure: a sulfur atom (S) is bonded to a carbon atom (C) that is ... attached to a chiral carbon ([C@H]).

Recovering Prompt

```
Instruction:
    C Methane (CH4)
    CC Ethane (CH3CH3)
    C=C Ethene (CH2CH2)
    C#C Ethyne (CHCH)
    COC Dimethyl ether (CH3OCH3)
    CC=O Acetaldehyde (CH3-CH=O)
    ...
```

```
Template:
  {
    "Summary" = {
        (S)-amphetamine is a 1-phenylpropan-2-amine
        that has S configuration. It has a role as a
        neurotoxin.
    }

    "Isomeric SMILE" =
    {   C[C@@H](CC1=CC=CC=C1)N

    }
  }
```

Query Generated Summary s_i:

The provided Isomeric SMILES represents a compound with the following structure: a sulfur atom (S) is bonded to a carbon atom (C) that is ... attached to a chiral carbon ([C@H]).

Fig. 7: PubMed dataset: A case of the generating-recovering annotation.

capabilities of GPT models to produce concise summaries and alleviate the burden of time and specialized expertise required by human annotators when dealing with complex structured data, such as graphs. Our approach consists of two phases: template optimization and label generation. During the template optimization stage, a candidate set and a validation set are sampled from the training data, a template is selected from the candidate set and used as a prompt to generate a textual summary using GPT models, and the same model is utilized to recover the original data from the generated summary, with alignment scores being calculated for feedback and potential template modification. Both sentence-level and structure-level alignment scores are assessed between the original and recovered data, demonstrating that our approach consistently achieves competitive evaluation scores.

References

[1] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[4] H. Laga, L. V. Jospin, F. Boussaid, and M. Bennamoun, "A survey on deep learning techniques for stereo-based depth estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 4, pp. 1738–1764, 2020.

[5] K.-F. Yang, C. Cheng, S.-X. Zhao, H.-M. Yan, X.-S. Zhang, and Y.-J. Li, "Learning to adapt to light," *International Journal of Computer Vision*, pp. 1–20, 2023.

[6] L. Cai, N. E. McGuire, R. Hanlon, T. A. Mooney, and Y. Girdhar, "Semi-supervised visual tracking of marine animals using autonomous underwater vehicles," *International Journal of Computer Vision*, pp. 1–22, 2023.

[7] H. Chen, H. Shi, X. Liu, X. Li, and G. Zhao, "Smg: A micro-gesture dataset towards spontaneous body gestures for emotional stress state analysis," *International Journal of Computer Vision*, vol. 131, no. 6, pp. 1346–1366, 2023.

[8] J. Kopp, D. Kellner, A. Piroli, and K. Dietmayer, "Tackling clutter in radar data–label generation and detection using pointnet++," *arXiv preprint arXiv:2303.09530*, 2023.

[9] A. P. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das, "Totto: A controlled table-to-text generation dataset," *arXiv preprint arXiv:2004.14373*, 2020.

[10] R. A. McEver, B. Zhang, C. Levenson, A. Iftekhar, and B. Manjunath, "Context-driven detection of invertebrate species in deep-sea video," *International Journal of Computer Vision*, vol. 131, no. 6, pp. 1367–1388, 2023.

[11] X. Yang, T. Burghardt, and M. Mirmehdi, "Dynamic curriculum learning for great ape detection in the wild," *International Journal of Computer Vision*, pp. 1–19, 2023.

[12] Y. Chen, Y. Peng, K.-W. Chang, M. Dredze, A. M. Cohen, W. R. Hersh, I. J. Marshall, A. Névéol, P. Zweigenbaum, S. Liu, B. Hu, F. Li, and Z. Lu, "Challenges and opportunities in automated coding of diagnosis and procedure in healthcare," *npj Digital Medicine*, vol. 4, no. 1, Nov. 2021. [Online]. Available: https://doi.org/10.1038/s41746-021-00520-6

[13] J. Zhang, M.-Y. Kan, K. Sugiyama, and T.-S. Chua, "Scientific document processing: challenges for modern learning methods," *International Journal on Digital Libraries*, vol. 32, no. 2, pp. 1–38, May 2023. [Online]. Available: https://doi.org/10.1007/s00799-023-00352-7

[14] G. C. Dobre, M. Gillies, and X. Pan, "Immersive machine learning for social attitude detection in virtual reality narrative games," *Springer*, 2022. [Online]. Available: http://dx.doi.org/10.1007/s10055-022-00644-4

[15] J. Deriu, A. Rodrigo, A. Otegi, G. Echegoyen, S. Rosset, E. Agirre, and M. Cieliebak, "Survey on evaluation methods for dialogue systems," *Springer*, 2021. [Online]. Available: http://dx.doi.org/10.1007/s10462-020-09866-x

[16] B. Ding, C. Qin, L. Liu, L. Bing, S. Joty, and B. Li, "Is gpt-3 a good data annotator?" *arXiv preprint arXiv:2212.10450*, 2022.

[17] F. Gilardi, M. Alizadeh, and M. Kubli, "Chatgpt outperforms crowd-workers for text-annotation tasks," *arXiv preprint arXiv:2303.15056*, 2023.

[18] Y. Zhu, P. Zhang, E.-U. Haq, P. Hui, and G. Tyson, "Can chatgpt reproduce human-generated labels? a study of social computing tasks," *arXiv preprint arXiv:2304.10145*, 2023.

[19] T. Labruna, S. Brenna, A. Zaninello, and B. Magnini, "Unraveling chatgpt: A critical analysis of ai-generated goal-oriented dialogues and annotations," *arXiv preprint arXiv:2305.14556*, 2023.

[20] B. Lester, N. Constant, and R. Al-Rfou, "The power of scale for parameter-efficient prompt tuning," in *EMNLP*, 2021.

[21] ——, "Guiding frozen language models with learned soft prompts," Google AI Blog, 2022. [Online]. Available: https://ai.googleblog.com/2022/02/guiding-frozen-language-models-with.html

[22] N. Polyzotis and M. Zaharia, "What can data-centric ai learn from data and ml engineering?" *arXiv preprint arXiv:2112.06439*, 2021.

[23] M. Zhang and J. Li, "A commentary of gpt-3 in mit technology review 2021," *Fundamental Research*, vol. 1, no. 6, pp. 831–833, 2021.

[24] B. D. Lund and T. Wang, "Chatting about chatgpt: how may ai and gpt impact academia and libraries?" *Library Hi Tech News*, vol. 40, no. 3, pp. 26–29, 2023.

[25] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg *et al.*, "Sparks of artificial general intelligence: Early experiments with gpt-4," *arXiv preprint arXiv:2303.12712*, 2023.

[26] P. Lee, S. Bubeck, and J. Petro, "Benefits, limits, and risks of gpt-4 as an ai chatbot for medicine," *New England Journal of Medicine*, vol. 388, no. 13, pp. 1233–1239, 2023.

[27] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, "Capabilities of gpt-4 on medical challenge problems," *arXiv preprint arXiv:2303.13375*, 2023.

[28] B. Peng, C. Li, P. He, M. Galley, and J. Gao, "Instruction tuning with gpt-4," *arXiv preprint arXiv:2304.03277*, 2023.

[29] J. Yang, H. Jin, R. Tang, X. Han, Q. Feng, H. Jiang, B. Yin, and X. Hu, "Harnessing the power of llms in practice: A survey on chatgpt and beyond," *arXiv preprint arXiv:2304.13712*, 2023.

[30] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.

[31] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *International conference on learning representations*, 2016.

[32] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," *Advances in neural information processing systems*, vol. 29, 2016.

[33] Y. Song, T. Wang, P. Cai, S. K. Mondal, and J. P. Sahoo, "A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities," *ACM Computing Surveys*, 2023.

[34] S. Jadon, "An overview of deep learning architectures in few-shot learning domain," *arXiv preprint arXiv:2008.06365*, 2020.

[35] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang, "Meta-baseline: Exploring simple meta-learning for few-shot learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9062–9071.

[36] O. Rubin, J. Herzig, and J. Berant, "Learning to retrieve prompts for in-context learning," *arXiv preprint arXiv:2112.08633*, 2021.

[37] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?" *arXiv preprint arXiv:2101.06804*, 2021.

[38] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, and Z. Sui, "A survey for in-context learning," *arXiv preprint arXiv:2301.00234*, 2022.

[39] Y. Hu, C.-H. Lee, T. Xie, T. Yu, N. A. Smith, and M. Ostendorf, "In-context learning for few-shot dialogue state tracking," *arXiv preprint arXiv:2203.08568*, 2022.

[40] J. Ye, Z. Wu, J. Feng, T. Yu, and L. Kong, "Compositional exemplars for in-context learning," *arXiv preprint arXiv:2302.05698*, 2023.

[41] X. Li and X. Qiu, "Finding supporting examples for in-context learning," *arXiv preprint arXiv:2302.13539*, 2023.

[42] M. Cliche, "Bb_twtr at semeval-2017 task 4: Twitter sentiment analysis with cnns and lstms," *arXiv preprint arXiv:1704.06125*, 2017.

[43] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano, "Learning to summarize with human feedback," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3008–3021, 2020.

[44] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.

[45] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out: Proceedings of the ACL-04 workshop*, vol. 8, 2004.

[46] K. Ganesan, "Rouge 2.0: Updated and improved measures for evaluation of summarization tasks," *arXiv preprint arXiv:1803.01937*, 2018.

[47] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[48] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[49] K. Kheiri and H. Karimi, "Sentimentgpt: Exploiting gpt for advanced sentiment analysis and its departure from current machine learning," *arXiv preprint arXiv:2307.10234*, 2023.