GAVx at SemEval-2024 Task 10: Emotion Flip Reasoning via Stacked Instruction Finetuning of LLMs

Vy Nguyen[⋄], **Xiuzhen Zhang**[†]
[⋄]School of Science, Engineering & Technology, RMIT University
[†]School of Computing Technologies, RMIT University
[⋄]s3964786@rmit.edu.vn, [†]xiuzhen.zhang@rmit.edu.au

Abstract

The Emotion Flip Reasoning task at SemEval 2024 aims at identifying the utterance(s) that trigger a speaker to shift from an emotion to another in a multi-party conversation. The spontaneous, informal, and occasionally multilingual dynamics of conversations make the task challenging. In this paper, we propose a supervised stacked instruction-based framework to finetune large language models to tackle this task. Utilising the annotated datasets provided, we curate multiple instruction sets involving chain-of-thoughts, feedback, and self-evaluation instructions, for a multi-step finetuning pipeline. We utilise the self-consistency inference strategy to enhance prediction consistency. Experimental results reveal commendable performance, achieving mean F1 scores of 0.77 and 0.76 for triggers in the Hindi-English and English-only tracks respectively. This led to us earning the second highest ranking[1] in both tracks.

1 Introduction

The EDiReF shared task at SemEval 2024 (Kumar et al., 2024) encompasses two challenges in natural language processing (NLP): *Emotion Recognition in Conversation* (ERC) and *Emotion Flip Reasoning* (EFR). Our work focuses on the latter challenge—EFR, which aims at identifying the utterances responsible for triggering a shift in a speaker's emotional state, hereafter referred to as an *emotion flip*, within a multi-party conversation. The task offers two tracks: one involving Hindi-English code-mixed conversations and the other focusing on English-only conversations. The first track particularly addresses the complexities inherent in multilingual contexts. Each track comes with its respective dataset annotated by the organisers, wherein each utterance is labelled to represent one of the six primary emotional states—*anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise* (Ekman, 1999). Additionally, the emotion *neutral* is assigned to utterances devoid of any expressed emotion (Kumar et al., 2024). Given these emotion labels, locating the emotion flip is straightforward. Our task is to identify the triggers behind it.

Figure 1 shows a Hindi-English code-mixed conversation conducted between two speakers, complemented by English translations. During the chat, Speaker A undergoes an emotion flip from *sadness* to *joy*, while Speaker B transitions from *surprise* to *joy*. Utterance *u4* is identified as the trigger causing both speakers' emotion flip. Particularly, when Speaker B delivers utterance *u4*, their emotional state also undergoes a change, rendering *u4* as a self-trigger for Speaker B's own emotion flip. It is worth noting that, for an emotion flip, there can be no trigger utterances at all, or there can be one or multiple trigger utterances originating from any involved speakers, including themselves (Kumar et al., 2024).

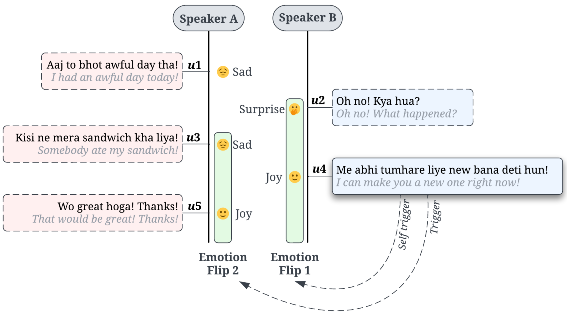

Speaker A
Speaker B
Aaj to bhot awful day tha!
I had an awful day today!
u1
Sad
Surprise
u2
Oh no! Kya hua?
Oh no! What happened?
Kisi ne mera sandwich kha liya!
Somebody ate my sandwich!
u3
Sad
Joy
u4
Me abhi tumhare liye new bana deti hun!
I can make you a new one right now!
Wo great hoga! Thanks!
That would be great! Thanks!
u5
Joy
Self trigger
Trigger
Emotion Flip 2
Emotion Flip 1

Figure 1: A conversation with five utterances between two speakers involving two emotion flips. Translations are not part of the conversation.

[1] https://codalab.lisn.upsaclay.fr/competitions/16769

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 326–336
June 20-21, 2024 ©2024 Association for Computational Linguistics

The EFR task can be formulated as follows: Given a conversation between p speakers $s_i^{i=1..p}$ involving q utterances $u_j^{j=1..q}$, each assigned an emotion $e_j^{j=1..q}$, if speaker s_i changes their emotion at utterance u_k, there may exist a set of utterances u_l, wherein $1 \leq l \leq k$, that trigger the emotion flip. If we use 1 to denote a *trigger utterance* and 0 to denote a *non-trigger utterance*, then the array $[t_1, t_2, \ldots, t_k]$, in which t_m equals either 0 or 1 and its position in the array corresponds to the position of the utterance in the conversation, can conveniently represent the task's label for an emotion flip. For instance, considering the conversation in Figure 1, the array [0, 0, 0, 1, 0] indicates that the utterance at position 4 caused Speaker A to shift from *sadness* to *joy*.

In this paper, we introduce an instruction-based framework designed to finetune large language models (LLMs) for addressing the EFR task. Initially, leveraging the training data, we construct multiple distinct instruction sets to guide the model in identifying triggers for emotion flips. These instructs emulate human cognitive processes, incorporating both human feedback and self-evaluation procedures as integral components of the reasoning process. Subsequently, we execute a supervised stacked finetuning pipeline to refine the model using these instructions. Once the model is tuned, we employ an inference strategy called *self-consistency* (Wang et al., 2023) to generate predictions for the test data.

Besides the system description, we made the following observations in our experiments:

1. Our framework demonstrates competent performance for both English-only and Hindi-English code-mixed datasets, indicating its capacity to effectively handle both monolingual and multilingual contexts.
2. Providing high-quality instructions to LLMs is crucial for achieving the desired output. Our model's performance improves each time we provide more refined instructions.
3. The self-consistency inference strategy helps mitigate the randomness in the output generated by LLMs, allowing us to attain more uniform results across executions.

In the next section, we discuss various related works. Subsequently, we detail our proposed system in Section 3. Following this, we outline our experimental setup in Section 4, analyse its results in Section 5 before concluding in Section 6.

2 Related Work

The EFR task was first introduced by Kumar et al. (2022), who utilised a masked memory network and a transformer-based architecture to tackle it. In subsequent research in 2023, they delved deeper into the instigators behind emotion flips and introduced a neural architecture named TGIF. This architecture integrates transformer encoders and stacked gated recurrent units (GRUs) to comprehensively capture the conversation context, speaker dynamics, and emotional sequences.

While EFR remains a relatively recent task, it is closely related to the widely studied task of *Emotion-Cause Pair Extraction* (ECPE) (Kumar et al., 2022). The objective of ECPE is to identify a text span that elicit a specific emotion (Xia and Ding, 2019). Earlier endeavours to address ECPE using deep learning faced challenges associated with position bias (Ding and Kejriwal, 2020). Zheng et al. (2022) introduced UECA-Prompt, a BERT-based universal prompt tuning method. Subsequently, Wu et al. (2024) proposed the DECC framework, which incorporates inducing inference and logical pruning to guide LLMs to reason. Both prompt-based approaches outperformed previous works on this task. The promising results observed in ECPE using prompt-based methods motivates us to adapt them to the EFR task.

Prompt-based learning refers to prompting pre-trained language models to tackle downstream tasks (Liu et al., 2021). Recently, LLMs like GPT (OpenAI, 2023) and LLAMA (Touvron et al., 2023) demonstrate exceptional performance across various NLP tasks, even with zero-shot or few-shot prompts (Brown et al., 2020; Sun et al., 2023). Several prompting techniques have emerged recently. Chain-of-thoughts (CoT) prompting, one of the most popular techniques, replicates human cognitive process by integrating intermediate reasoning steps (Wei et al., 2023). Instead of attempting to reach the answer in a single leap, this approach encourages the model to divide complicated problems into smaller, more manageable components, imitating the way humans think. Tree-of-thoughts prompting extends CoT by constructing a tree of logical steps towards the solution (Yao et al., 2023). Multimodal CoT combines text and vision into a two-phase cognitive process (Zhang et al., 2023). On the other hand, instead of fixed prompts, LLMs themselves can be used to dynamically generate prompts for downstream tasks (Zhou et al., 2022) or to produce and execute programming code as intermediate steps (Gao et al., 2022). Interestingly, LLMs are demonstrated to be capable of generating and analysing recursive reasoning, a unique cognitive ability akin to human thinking processes (Dąbkowski and Beguš, 2023).

Despite these emerging techniques, LLMs often generate outputs that deviate from the ground truth labels (Wadhwa et al., 2023). To address this challenge, instruction tuning emerges as a solution, employing supervised learning on a set of instructions to narrow the gap between the output generated by the base LLMs and the desired output (Zhang et al., 2023). Additionally, human and augmented feedback play a crucial role in mitigating this issue. Akyurek et al. (2023) introduced a reinforcement learning framework equipped with a critique generator to guide GPT-3 in improving its output. Diao et al. (2023) proposed the Active-Prompt method, which entails human manual annotation of uncertain rational chains. Furthermore, Paranjape et al. (2023) devised a novel framework that freezes LLMs and integrates reasoning steps from an external program.

These prior studies underscore the significance of furnishing high-calibre instructions and feedback, as well as employing suitable prompting techniques, to achieve the desired output with LLMs.

3 Our System

In this section, we describe the general approach and the implementation of our system.

3.1 General Approach

Our system must be built upon an *instruction tuneable LLM*. The approach involves two stages: instruction tuning and inference.

3.1.1 Instruction Tuning

Our approach is founded on the premise that problems necessitating reasoning often allows multiple reasoning paths to arrive at the same correct solution. To instil the desired reasoning capabilities in an LLM, we adopt a supervised tuning approach using instructions derived from the training data (Zhang et al., 2023) and implement a stacked framework employing diverse instruction sets to foster the model's ability in navigating varied reasoning paths. A summary of each step is provided below.

Step 1. We train the base model with *Chain-of-thoughts (CoT) instructions*. These instructions can be generated from the training data. This step trains the model on *what is right*.

Step 2. We further provide *feedback-based instructions* to tuned model, expecting it to *rectify the discrepancy* between its current reasoning manner and the desired reasoning manner.

Step 3. We further provide *self-evaluation instructions* to the tuned model, expecting it to enhance its ability to improve itself through *autonomous* evaluation.

Figure 2 summarises the main steps in this supervised finetuning pipeline. Section 3.3.1 describes how we construct these instruction sets.

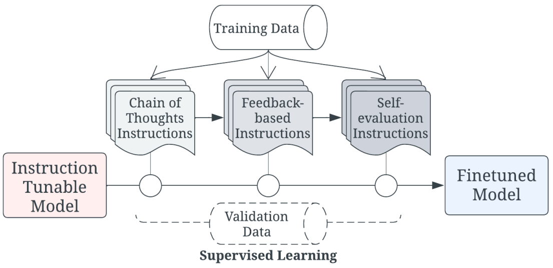

Training Data
Chain of Thoughts Instructions
Feedback-based Instructions
Self-evaluation Instructions
Instruction Tunable Model
Finetuned Model
Validation Data
Supervised Learning

Figure 2: Supervised instruction tuning pipeline.

3.1.2 Inference Procedure

Language models are susceptible to random errors in reasoning, potentially resulting in incorrect conclusions (Wang et al., 2023). To mitigate this issue, these researchers introduced the *self-consistency* (SC) inference strategy. It operates on the principle of generating diverse reasoning paths and selecting the most consistent conclusion by marginalising any inconsistent ones. We adapt this inference strategy to align with the characteristics of our own model.

In our tailored version of SC, we iteratively prompt the model with a progressively increasing *temperature*, a parameter controlling the randomness of the output (Wang et al., 2020), until the answers converge. We introduce a threshold α to determine the convergence point. The convergence condition is if the average of the predicted labels for an utterance is not less than α or not greater than $1 - \alpha$. Once the answers converge, the final label for each utterance is the average prediction rounded to the nearest integer, which is eventually either 0.0 or 1.0. Besides the α threshold, we also impose a minimum and maximum number of prompts so that sufficient runs are performed while ensuring the algorithm still stops if it does not converge. Figure 3 presents a piece of pseudo-code for this inference strategy.

```
Global variables: min, max, α
Converged(prediction):
  If each item in prediction array ≥ α or ≤ 1 − α:
    Return True
  Return False
Infer(prediction, dialogue, temp, count):
  Prompt for dialogue, with temperature as temp
  Assign result array to new_prediction
  Set prediction to mean of all predictions so far
  Increase count by 1
  Increase temp by a constant
  If (max ≤ count) or
     (min ≤ count and Converged(prediction)):
    Return round(prediction)
  Return Infer(prediction, dialog, temp, count)
```

Figure 3: Our tailored version of SC.

3.2 System Implementation

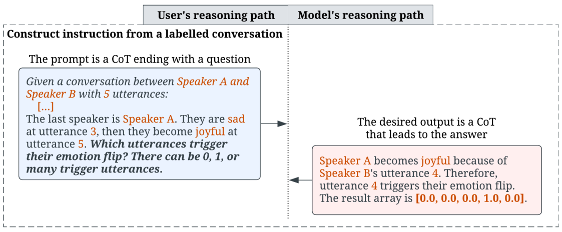
In this section, we describe how we implement the framework we conceptualise above.

3.2.1 Instruction Construction

We use the training data to construct the instruction sets. Each instruction comprises two components: a *prompt* and a *desired output*. Our finetuning pipeline requires three different instruction sets to be built as follows.

CoT instruction set—The *prompt* includes a labelled conversation sampled from the training data, a CoT that describes the progression of emotional states for the last speaker, and a query tasking the model with identifying the triggers. The *desired output* is a CoT that leads to the accurate identification. We programmatically generate these instructions using a dynamic text template that outlines the sequence of reasoning. The template contains placeholders that can be populated with matching information derived from the conversation. Figure 4 shows how a CoT instruction is crafted for a typical conversation, where each utterance originates from a single speaker, an emotion flip trigger is present, and it is not a self-trigger. Our implementation of the text template is versatile, capable of accommodating various scenarios, including those with no triggers, self-triggers, multiple triggers, and instances where an utterance is attributed to multiple or all speakers.

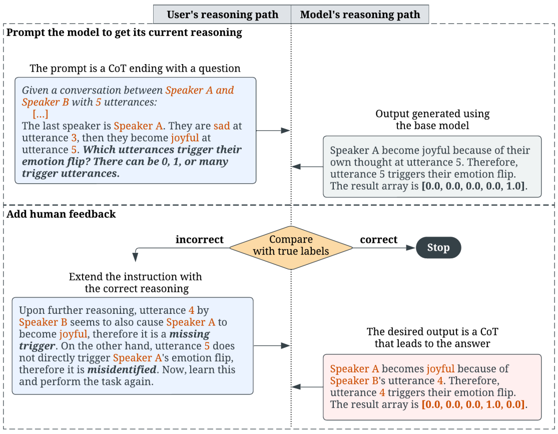
Feedback-based instruction set—The *prompt* is constructed by sampling a labelled conversation and asking the model to identify the emotion flip triggers directly. Subsequently, its output is then compared with the ground truth labels. If discrepancies arise, the *prompt* is extended with feedback regarding missed or misidentified triggers, and a request for the model to retry the task. We utilise the model tuned using CoT instructions for this step, which enables us to assess its current reasoning manner. Following this, the *desired output* is a CoT that leads to the correct answer. Figure 5 provides an overview of constructing a feedback-based instruction through the integration of a labelled conversation and a baseline model. In our implementation, we again employ dynamic text templates to generate the prompt and desired output for various scenarios, including instances where multiple triggers are missed or misidentified, all triggers are misidentified, and self-triggers are misidentified.


User's reasoning path
Model's reasoning path
Construct instruction from a labelled conversation
The prompt is a CoT ending with a question
The desired output is a CoT that leads to the answer

Given a conversation between Speaker A and Speaker B with 5 utterances:
[...]
The last speaker is Speaker A. They are sad at utterance 3, then they become joyful at utterance 5. ***Which utterances trigger their emotion flip? There can be 0, 1, or many trigger utterances.***

Speaker A becomes joyful because of Speaker B's utterance 4. Therefore, utterance 4 triggers their emotion flip. The result array is **[0.0, 0.0, 0.0, 1.0, 0.0]**.

Figure 4: The construction of a CoT instruction for a conversation. Texts in colour indicate placeholders.


User's reasoning path
Model's reasoning path
Prompt the model to get its current reasoning
The prompt is a CoT ending with a question
Output generated using the base model
Add human feedback
incorrect
Compare with true labels
correct
Stop
Extend the instruction with the correct reasoning
The desired output is a CoT that leads to the answer

Given a conversation between Speaker A and Speaker B with 5 utterances:
[...]
The last speaker is Speaker A. They are sad at utterance 3, then they become joyful at utterance 5. ***Which utterances trigger their emotion flip? There can be 0, 1, or many trigger utterances.***

Speaker A become joyful because of their own thought at utterance 5. Therefore, utterance 5 triggers their emotion flip. The result array is **[0.0, 0.0, 0.0, 0.0, 1.0]**.

Upon further reasoning, utterance 4 by Speaker B seems to also cause Speaker A to become joyful, therefore it is a ***missing trigger***. On the other hand, utterance 5 does not directly trigger Speaker A's emotion flip, therefore it is ***misidentified***. Now, learn this and perform the task again.

Speaker A becomes joyful because of Speaker B's utterance 4. Therefore, utterance 4 triggers their emotion flip. The result array is **[0.0, 0.0, 0.0, 1.0, 0.0]**.

Figure 5: The construction of a feedback-based instruction for a conversation. Texts in colour indicate placeholders.

Self-evaluation instruction set—The *prompt* is structured similarly to a feedback-based instruction, involving the selection of a labelled conversation, and prompting the model finetuned in Step 2 to replicate its current reasoning approach. However, in cases where the output is inaccurate, the prompt extends to instruct the model to evaluate its own output and iteratively retry the task until satisfaction is achieved. The *desired output* is an augmented CoT that emulates a recursive reasoning and evaluation process, culminating in the correct answer. Our approach to constructing self-evaluation instructions is inspired by research indicating that LLMs possess recursive reasoning abilities (Dąbkowski and Beguš, 2023). Leveraging this capability, we instruct LLMs to engage in autonomous evaluation. To implement this idea, we compile a dynamic text template to simulate a recursive thinking process with information extracted from the given conversation. This template enables the generation of a variable number of iterations, mirroring the iterative cognitive process observed in humans, which may not always yield perfect results in the initial iterations. Figure 6 illustrates the construction of the prompt and the simulation of an expected output using two reasoning iterations before reaching a correct answer.

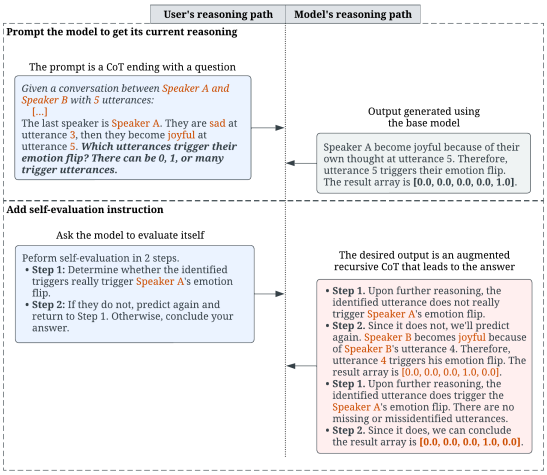

User's reasoning path
Model's reasoning path
Prompt the model to get its current reasoning
The prompt is a CoT ending with a question
Given a conversation between Speaker A and Speaker B with 5 utterances:
[...]
The last speaker is Speaker A. They are sad at utterance 3, then they become joyful at utterance 5. Which utterances trigger their emotion flip? There can be 0, 1, or many trigger utterances.
Output generated using the base model
Speaker A become joyful because of their own thought at utterance 5. Therefore, utterance 5 triggers their emotion flip. The result array is [0.0, 0.0, 0.0, 0.0, 1.0].
Add self-evaluation instruction
Ask the model to evaluate itself
Peform self-evaluation in 2 steps.
• Step 1: Determine whether the identified triggers really trigger Speaker A's emotion flip.
• Step 2: If they do not, predict again and return to Step 1. Otherwise, conclude your answer.
The desired output is an augmented recursive CoT that leads to the answer
• Step 1. Upon further reasoning, the identified utterance does not really trigger Speaker A's emotion flip.
• Step 2. Since it does not, we'll predict again. Speaker B becomes joyful because of Speaker B's utterance 4. Therefore, utterance 4 triggers his emotion flip. The result array is [0.0, 0.0, 0.0, 1.0, 0.0].
• Step 1. Upon further reasoning, the identified utterance does trigger the Speaker A's emotion flip. There are no missing or missidentified utterances.
• Step 2. Since it does, we can conclude the result array is [0.0, 0.0, 0.0, 1.0, 0.0].

Figure 6: The construction of a self-evaluation instruction for a conversation. Texts in colour indicate placeholders.

As the building of the feedback-based and the self-evaluation instruction sets requires the model to undergo learning from the preceding step, our system must be finetuned in a sequential pipeline.

3.2.2 Prompting Finetuned Model

Following the finetuning of the base LLM with the three prepared instruction sets, we proceed with the SC inference procedure to make predictions for unlabelled data. A critical aspect of this process involves prompting the finetuned model in diverse manners to elicit varied reasoning paths. Given the utilisation of three instruction sets, we employ three distinct prompt variants to prompt the model in identifying emotion flip triggers. The prompt variants utilised are detailed in Figure 7.

Extracting the label from the output sequence generated by the model requires engineering effort due to the dynamics of LLMs. When multiple labels exist in the output, our implementation selects the last label. This aligns with our tuning technique, where intermediate predictions may undergo adjustments during subsequent re-evaluations.

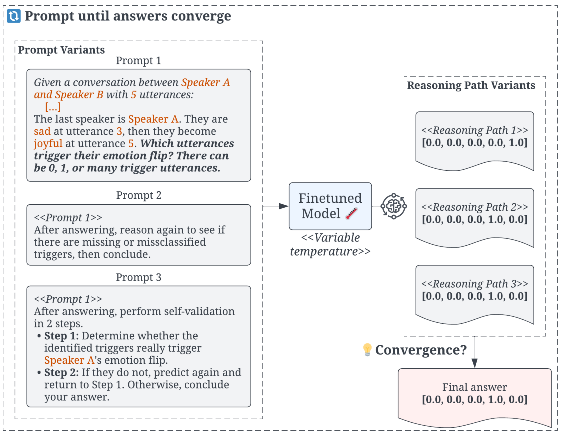

Prompt until answers converge
Prompt Variants
Prompt 1
Given a conversation between Speaker A and Speaker B with 5 utterances:
[...]
The last speaker is Speaker A. They are sad at utterance 3, then they become joyful at utterance 5. Which utterances trigger their emotion flip? There can be 0, 1, or many trigger utterances.
Prompt 2
<<Prompt 1>>
After answering, reason again to see if there are missing or missclassified triggers, then conclude.
Prompt 3
<<Prompt 1>>
After answering, perform self-validation in 2 steps.
• Step 1: Determine whether the identified triggers really trigger Speaker A's emotion flip.
• Step 2: If they do not, predict again and return to Step 1. Otherwise, conclude your answer.
Finetuned Model
<<Variable temperature>>
Reasoning Path Variants
<<Reasoning Path 1>> [0.0, 0.0, 0.0, 0.0, 1.0]
<<Reasoning Path 2>> [0.0, 0.0, 0.0, 1.0, 0.0]
<<Reasoning Path 3>> [0.0, 0.0, 0.0, 1.0, 0.0]
Convergence?
Final answer [0.0, 0.0, 0.0, 1.0, 0.0]

Figure 7: Multiple prompts variants are utilised to produce varied reasoning paths.

4 Experimental Setup

4.1 Datasets

In our experiments, we utilise the datasets provided by the organisers. The data for both tracks originate from previously published datasets. The Hindi-English dataset is sourced from MaSaC, a multimodal dataset compiled from the Hindi TV series *Sarabhai vs Sarabhai* (Bedi et al., 2023). The English monolingual dataset is sourced from MELD, a dataset containing dialogues from the American TV sitcom *Friends* (Poria et al., 2019). In the Hindi-English track, a new emotion, *contempt*, appears, which does not impact our approach since it solely focuses on the positions of the utterance before and after the emotion flip, not the emotions themselves. Table 1 summarises the shape of both datasets.

Upon closer examination of the training splits, it is evident that a significant portion of triggers lies within a proximity of either 1 or 2 utterances preceding the emotion flip. Furthermore, conversations in the Hindi-English dataset exhibit greater length and involve more speakers on average compared to the English-only dataset.

Statistics on the training splits for both tracks are shown in Table 2.

Split	Instances	Utterances	Triggers	% Triggers
Hindi-English dataset				
Train	4,894	98,777	6,542	6.62%
Val	3,89	7,462	434	5.82%
Test	3,85	7,690	461	5.99%
English-only dataset				
Train	4,000	35,000	5,575	15.93%
Val	426	3,522	494	14.03%
Test	1,002	8,642	1,169	13.53%

Table 1: Shape of the datasets provided.

	Utterances	Triggers	Speakers	Distance
Hindi-English dataset				
Min	1.00	0.00	1.00	0.00
Mean	20.19	1.34	3.59	1.43
75%	27.00	2.00	4.00	2.00
Max	106.00	6.00	10.00	21.00
English-only dataset				
Min	2.00	0.00	1.00	0.00
Mean	8.75	1.39	2.62	1.38
75%	12.00	2.00	3.00	1.00
Max	24.00	12.00	8.00	17.00

Table 2: Statistics on the training splits.

4.2 Evaluation Method

We utilise the F1 score of the identified trigger utterances, labelled as 1, as the primary evaluation metric. The F1 score, which balances precision and recall, serves as a robust metric to evaluate the model's ability to accurately identify emotion flip triggers while considering both false positives and false negatives (Goutte and Gaussier, 2005).

To assess the efficacy of our system, we establish a baseline by referring to the results obtained using masked memory networks and transformers by the researchers who proposed the EFR task (Kumar et al. 2022). Subsequently, we conduct an ablation study, aiming to discern the impact of each component in the architecture on the overall performance of the model. Furthermore, we also perform cross-lingual inference to assess the cross-lingual capability of our approach.

4.3 Tuning and Inference Settings

We use the model GPT-3.5-Turbo-1106 by OpenAI[2] as the base model and Azure[3] as the infrastructure. For each track, we separately finetune the model in five epochs using a batch size of 8 and a learning rate multiplier of 1.0, while also incorporating a prompt loss weight. Due to the impromptu and informal nature of conversations, a low content filter setting is consistently used throughout all stages so that the model accepts more contents in their original form.

After finetuning, we generate predictions for the test data using the SC inference strategy. We incorporate a minimum of 3 prompts and a maximum of 10 prompts, alongside an α threshold set at 0.75. This stringent threshold dictates that a consensus of at least 3 out of 4 (75%) agreement amongst predicted labels for an utterance must be achieved before the final label is determined. Furthermore, the temperature parameter is initialised at 0.0 and progressively incremented by 0.1 in each iteration. This iterative adjustment facilitates the introduction of increasing randomness into the model's output, thereby mitigating the risk of overfitting.

5 Results and Analysis

5.1 Main results

In this section, we conduct five executions for each test and report the averages to obtain reliable results. Table 3 provides a summary of the models' performance across all tests conducted.

Initially, to evaluate the base LLMs, we conduct one-shot prompting using the GPT-3.5-Turbo-1106 and GPT-4-0613 models. This prompt construction mirrors that of the CoT instruction set. Our results reveal that GPT-4-0613 achieves an F1 score of 0.60 for the English-only track, surpassing the baseline by 0.061 points, without prior training. Similarly, it shows comparable performance in the Hindi-English track, achieving an F1 score of 0.57.

Subsequent tests demonstrate that finetuning the base GPT-3.5-Turbo-1106 model with additional instructions consistently enhances its performance. We utilise a distinct prompt variant at each tuning stage for zero-shot prediction to prompt the model to reason according to our desired approach, as described in Section 3.3.1. We then apply the SC procedure on the fully tuned model. Integrating all proposed techniques into the final model yields a plateau F1 score of 0.77 and 0.76 for the Hindi-English and English-only tracks respectively. Note

[2] `https://platform.openai.com/docs/models`

[3] `https://azure.microsoft.com`

that in the SemEval-2024 Task 10 leader board, we achieved 0.79 for the former track, which was our best run. The results reported in this paper are the mean F1 scores across five runs.

System	Prompt	Accuracy	F1 (0)	F1 (1)
Hindi-English track				
GPT-3.5-Turbo-1106	1-shot	0.95	0.97	0.53
GPT-4-0613	1-shot	0.95	0.97	0.57
GPT-3.5 tuned Step 1	0-shot	0.96	0.98	0.67
GPT-3.5 tuned Step 2	0-shot	0.97	0.98	0.71
GPT-3.5 fully tuned	0-shot	0.97	0.99	0.73
GPT-3.5 fully tuned	SC	0.98	0.99	**0.77**
English-only track				
GPT-3.5-Turbo-1106	1-shot	0.88	0.93	0.57
GPT-4-0613	1-shot	0.89	0.93	0.60
GPT-3.5 tuned Step 1	0-shot	0.91	0.95	0.69
GPT-3.5 tuned Step 2	0-shot	0.92	0.96	0.72
GPT-3.5 fully tuned	0-shot	0.93	0.96	0.74
GPT-3.5 fully tuned	SC	0.95	0.96	**0.76**

Table 3: Model performance in different settings.

5.2 Error Analysis

Our quantitative analysis indicates that the test data provided are representative of the training data. Table 4 shows the confusion matrices of the fully tuned models for both tracks. In the Hindi-English code-mixed track, the model exhibits a tendency to misclassify triggers as non-triggers. Conversely, in the English-only track, a notable balance exists between misidentified triggers and misidentified non-triggers, despite the class imbalance.

Upon closer examination, Table 5 displays the frequency of each type of emotional flip, along with the corresponding number of accurate predictions. In this table, a prediction for a conversation is considered accurate only when all triggers and non-triggers are correctly identified. The data shows that across both tracks, emotion flips from *neutral* to *joy* and from *joy* to *neutral* are the most prevalent. The model achieves accuracy rates of 67.27% and 70.16% in identifying the triggers for these flips in the Hindi-English and English-only tracks respectively.

5.3 Ablation Analysis

In our ablation analysis, we note a consistent improvement in model performance with the addition of each instruction set. Table 6 illustrates these findings, indicating that each successive step reduces the number of false positive and false negative errors from its previous step. Despite that, it also introduces new errors into the predictions; however, the number of new errors is consistently lower than the errors reduced. Notably, tuning the model with CoT instructions at Step 1 emerges as the most impactful, reducing error rates by 38% and 25%, thus increasing F1 scores by 0.15 and 0.12 points for the Hindi-English and English tracks respectively. This highlights the efficacy of instruction tuning. Even with only one instruction set, the disparity between the base model's reasoning manner and the desired reasoning manner is significantly diminished. Subsequent steps further diminish errors, ultimately resulting in the plateau performance observed when employing all techniques in conjunction.

Hindi-English track

Predicted \ *True Label*	0	1
0	**7,201**	113
1	73	**303**

English-only track

Predicted \ *True Label*	0	1
0	**7,197**	285
1	276	**884**

Table 4: Confusion matrices for the fully tuned models.

Hindi-English track

Emotion After \ *Emotion Before*	*Ag*	*Ct*	*Dg*	*Fr*	*Jy*	*Nt*	*Sn*	*Sp*
Ag		5_3	1_0	1_0	8_7	23_{15}	0_0	3_2
Ct	4_3		0_0	2_0	12_9	15_{10}	0_0	1_1
Dg	3_2	1_1		0_0	0_0	1_1	0_0	1_1
Fr	2_0	0_0	1_0		2_0	13_{11}	2_2	1_1
Jy	6_6	2_1	1_0	3_1		38_{22}	5_4	3_3
Nt	27_{21}	22_{14}	2_0	15_{14}	72_{52}		9_5	13_9
Sn	4_3	2_1	0_0	3_2	12_9	12_6		1_1
Sp	6_6	3_3	0_0	0_0	7_4	14_{14}	0_0	

English-only track

Emotion After \ *Emotion Before*	*Ag*	*Dg*	*Fr*	*Jy*	*Nt*	*Sn*	*Sp*
Ag		13_7	9_7	14_8	65_{35}	15_{10}	28_{19}
Dg	7_6		1_1	3_2	19_{10}	4_2	5_3
Fr	2_2	1_0		4_2	20_{13}	3_3	4_2
Jy	12_7	1_1	3_3		119_{75}	19_{14}	31_{18}
Nt	73_{55}	16_{13}	17_{14}	119_{92}		47_{40}	67_{54}
Sn	22_{12}	2_0	2_1	13_{10}	49_{32}		17_6
Sp	27_{18}	7_7	2_2	24_{19}	83_{66}	13_{11}	

Table 5: Statistics of the model's accurate predictions for each emotion flip. Cell values present the frequency for an emotion flip, while subscript values present the number of accurate predictions. Top 2 mostly seen flips are highlighted in grey. *Abbreviations:* Anger (Ag), Contempt (Ct), Disgust (Dg), Fear (Fr), Joy (Jy), Neutral (Nt), Sadness (Sn), and Surprise (Sp).

Error	GPT-3.5	+CoT	+Feedback	+Self-eval	+SC
Hindi-English track					
FP	223	113^{-130}_{+20}	105^{-18}_{+8}	89^{-29}_{+13}	73^{-22}_{+6}
FN	185	137^{-61}_{+13}	130^{-14}_{+7}	124^{-10}_{+4}	113^{-17}_{+6}
English-only track					
FP	583	414^{-194}_{+25}	351^{-81}_{+18}	313^{-47}_{+9}	276^{-30}_{+7}
FN	478	344^{-171}_{+37}	319^{-41}_{+16}	298^{-31}_{+10}	285^{-17}_{+4}

Table 6: Ablation analysis of the model performance at each stage. Superscript values indicate the number of errors reduced, while subscript values indicate the number of newly introduced errors. *Abbreviations:* False Positive (FP), False Negative (FN).

5.4 Effectiveness of SC Inference Strategy

Previous sections show that SC improves the F1 score for both tracks. This section proceeds to deep dive into this strategy. Table 7 shows a conversation excerpted from the test data between Mark and Rachel, wherein there exists no triggers for Rachel's emotion flip from *anger* to *neutral*, hence the ground truth label is [0, 0, 0]. This instance is tricky, as Mark's question, Rachel's response, and her subsequent exclamation all appear relevant to the emotion flip. With α set at 0.75, after the first three prompt variants, the model's outputs do not align. However, as we prompt with a progressively higher temperature, convergence is achieved after 8 iterations, with at least 75% of the predictions for each utterance now in agreement. As a result, the predicted label matches the true label. This example aptly illustrates the efficacy of SC in resolving disagreements between different reasoning paths.

Mark: *Why do all your coffee mugs have numbers on the bottom?* [***Surprise***]
Rachel: *Oh. That's so Monica can keep track. That way if one on them is missing, she can be like, "Where's number 27?!"* [***Anger***]
Rachel: *Y'know what?* [***Neutral***]

Iter	Prompt	Temp	Prediction	Running Average
1	1	0.0	[0, 1, 0]	[0.00, 1.00, 0.00]
2	2	0.1	[0, 0, 0]	[0.00, 0.50, 0.00]
3	3	0.2	[0, 0, 1]	[0.00, 0.33, 0.33]
4	1	0.3	[0, 0, 1]	[0.00, 0.25, 0.50]
5	2	0.4	[0, 0, 0]	[0.00, 0.20, 0.40]
6	3	0.5	[1, 0, 0]	[0.17, 0.17, 0.33]
7	1	0.6	[0, 0, 0]	[0.14, 0.14, 0.28]
8	2	0.7	[0, 1, 0]	**[0.14, 0.25, 0.25]**

Table 7: Efficacy of SC in helping resolve disagreements between different reasoning paths for a sample conversation excerpted from test data.

5.5 Cross-lingual Inference

To assess the cross-lingual generalisability of our approach, we use the model trained on the Hindi-English dataset to predict outcomes for the English-only track, and reciprocally, the model trained on the English-only track for the Hindi-English dataset. The results presented in Table 8 demonstrate that our models achieve commendable performance, both surpassing GPT-4-0613, despite not being finetuned on data representative of the test data provided.

Model	Test Data	Accuracy	F1 (0)	F1 (1)
Hindi-English	English-only	0.92	0.95	**0.69**
English-only	Hindi-English	0.96	0.98	**0.64**

Table 8: Model performance using cross-lingual inference.

5.6 Model Hallucination

When fine-tuning GPT-3.5, we encountered a peculiar form of hallucination—an instance where the model generates outputs that largely deviate from the provided training data (Ji et al., 2023). Despite being explicitly instructed to classify *each utterance* as '0' or '1', the model predictions include '2' for some utterances in one execution and include more labels than the number of utterances in another execution. We eventually decided to omit these anomalous executions to maintain the integrity of our results. Currently, controlling this type of hallucination remains a challenge. Further research is necessary to mitigate this phenomenon and improve the model's adherence to its operational constraints.

5.7 Other Constraints

In our experiments, we leverage GPT models hosted on Azure cloud infrastructure. While this offers convenience and efficiency, they are not without their associated costs. Our finetuning process demands 3.5 hours of training time, encompassing approximately 2 million training tokens alongside nearly 200,000 prompting tokens. Additionally, the SC strategy necessitates multiple prompts to attain convergence, thereby extending the time required to derive final predictions. With our settings, the average model speed is 1.22 seconds per prompt for the Hindi-English track and 0.83 seconds per prompt for the English-only track. In light of these considerations, it is crucial to diligently address cost and resource constraints when building the models.

6 Conclusion & Future Work

The paper presents an instruction based LLM finetuning framework to address the EFR task. Our strategy employs a multilayered finetuning pipeline, utilising three diverse instruction sets to steer the model towards recognising emotion flip triggers and, and finalised with the application of the SC inference strategy. The framework benefits significantly from the provision of high-quality instructions, as evidenced by the progressively improved performance of our model as better-quality feedback and instructions are incorporated into the finetuning pipeline. The robustness of our framework is demonstrated by its proficient handling of both English-only and Hindi-English code-mixed datasets, affirming its effectiveness in varied linguistic scenarios. Through these findings, we trust that our study makes a meaningful impact on the field of prompt-based learning techniques by harnessing the evolved capabilities of LLMs.

Moving forward, our focus will be on an in-depth exploration of various instruction types to devise the optimal way to amalgamate them for the most generalisability. Furthermore, we plan to develop a systematic method for constructing a processing pipeline tailored to this task and potentially applicable to related NLP tasks. This pipeline will be designed to encompass a CoT prompts, incorporate feedback mechanisms, and integrate self-evaluation instructions to ensure a robust, repeatable process for enhancing model performance.

References

Afra Feyza Akyurek, Ekin Akyurek, Ashwin Kalyan, Peter Clark, Derry Tanti Wijaya, and Niket Tandon. 2023. RL4F: Generating Natural Language Feedback with Reinforcement Learning for Repairing Model Outputs. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7716–7733, Toronto, Canada. Association for Computational Linguistics.

Manjot Bedi, Shivani Kumar, Md Shad Akhtar, and Tanmoy Chakraborty. 2023. Multi-Modal Sarcasm Detection and Humor Classification in Code-Mixed Conversations. *IEEE Transactions on Affective Computing*, 14(02):1363–1375.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, et al. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs].

Maksymilian Dąbkowski and Gašper Beguš. 2023. Large language models and (non-)linguistic recursion. arXiv:2306.07195 [cs].

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active Prompting with Chain-of-Thought for Large Language Models. arXiv:2302.12246 [cs].

Jiayuan Ding and Mayank Kejriwal. 2020. An Experimental Study of The Effects of Position Bias on Emotion Cause Extraction. arXiv:2007.15066 [cs].

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020. End-to-End Emotion-Cause Pair Extraction based on Sliding Window Multi-Label Learning. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3574–3583, Online. Association for Computational Linguistics.

Paul Ekman. 1992. An argument for basic emotions. *Cognition and Emotion*, 6(3–4):169–200.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided Language Models. arXiv:2211.10435 [cs].

Qingqing Gao, Biwei Cao, Xin Guan, Tianyun Gu, Xing Bao, Junyan Wu, Bo Liu, and Jiuxin Cao. 2022. Emotion recognition in conversations with emotion shift detection based on multi-task learning. *Knowledge-Based Systems*, 248:108861.

Cyril Goutte and Eric Gaussier. 2005. A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation. In David E. Losada and Juan M. Fernández-Luna, editors, *Advances in Information Retrieval*, pages 345–359, Berlin, Heidelberg. Springer.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. ACM Computing Surveys, 55(12):248:1-248:38.

Shivani Kumar, Shubham Dudeja, Md Shad Akhtar, and Tanmoy Chakraborty. 2023. Emotion Flip Reasoning in Multiparty Conversations. *IEEE Transactions on Artificial Intelligence*:1–10.

Shivani Kumar, Anubhav Shrimal, Md Shad Akhtar, and Tanmoy Chakraborty. 2022. Discovering emotion and reasoning its flip in multi-party conversations using masked memory network and transformer. *Knowledge-Based Systems*, 240:108112.

Shivani Kumar, Md Shad Akhtar, Erik Cambria, and Tanmoy Chakraborty. 2024. SemEval 2024 -- Task 10: Emotion Discovery and Reasoning its Flip in Conversation (EDiReF). arXiv:2402.18944 [cs].

Wei Li, Yang Li, Vlad Pandelea, Mengshi Ge, Luyao Zhu, and Erik Cambria. 2023. ECPEC: Emotion-Cause Pair Extraction in Conversations. *IEEE Transactions on Affective Computing*, 14(3):1754–

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs].

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, et al. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs].

Bhargavi Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Tulio Ribeiro. 2023. ART: Automatic multi-step reasoning and tool-use for large language models. arXiv:2303.09014 [cs].

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. MELD: A Multimodal Multi-Party Dataset for Emotion Recognition in Conversations. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy. Association for Computational Linguistics.

Xiaofei Sun, Linfeng Dong, Xiaoya Li, Zhen Wan, Shuhe Wang, Tianwei Zhang, Jiwei Li, Fei Cheng, Lingjuan Lyu, Fei Wu, and Guoyin Wang. 2023. Pushing the Limits of ChatGPT on NLP Tasks. arXiv:2306.09719 [cs].

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971 [cs].

Somin Wadhwa, Silvio Amir, and Byron C. Wallace. 2023. Revisiting Relation Extraction in the era of Large Language Models. arXiv:2305.05003 [cs].

Juntao Wang and Tsunenori Mine. 2023. Multi-Task Learning for Emotion Recognition in Conversation with Emotion Shift. In Chu-Ren Huang, Yasunari Harada, Jong-Bok Kim, Si Chen, Yu-Yin Hsu, Emmanuele Chersoni, Pranav A, Winnie Huiheng Zeng, Bo Peng, Yuxi Li, and Junlin Li, editors, *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation*, pages 257–266, Hong Kong, China. Association for Computational Linguistics.

Pei-Hsin Wang, Sheng-Iou Hsieh, Shih-Chieh Chang, Yu-Ting Chen, Jia-Yu Pan, Wei Wei, and Da-Chang Juan. 2020. Contextual Temperature for Language Modeling. arXiv:2012.13575 [cs].

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs].

Jialiang Wu, Yi Shen, Ziheng Zhang, and Longjun Cai. 2024. Enhancing Large Language Model with Decomposed Reasoning for Emotion Cause Pair Extraction. arXiv:2401.17716 [cs].

Rui Xia and Zixiang Ding. 2019. Emotion-Cause Pair Extraction: A New Task to Emotion Analysis in Texts. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012, Florence, Italy. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601 [cs].

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023a. Instruction Tuning for Large Language Models: A Survey. arXiv:2308.10792 [cs].

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2023b. Multimodal Chain-of-Thought Reasoning in Language Models. arXiv:2302.00923 [cs].

Xiaopeng Zheng, Zhiyue Liu, Zizhen Zhang, Zhaoyang Wang, and Jiahai Wang. 2022. UECA-

Prompt: Universal Prompt for Emotion Cause Analysis. In Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, et al., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 7031–7041, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large Language Models Are Human-Level Prompt Engineers. arXiv:2211.01910 [cs].