Internship weekly report

Mathis Embit

MILES, LAMSADE

June 6, 2024

Introduction

This week: comparison between our solution and FluentPrompt [Shi+22].

Outline

FluentPrompt

Contributions:

- helps bridge the gap between manual prompt engineering and gradient-based prompt tuning
- show that topic relatedness and calibration of the prompts are key to their success
- new method for discovering effective prompts without the need for labeled data

Why FluentPrompt works

FLUENTPROMPT generates a diverse human-readable prompts. Introducing noise helps getting diverse optimized prompts. This allows to explore the relationship between the features of the prompts and their performance.

To do so they Langevin dynamics: add a progressive Gaussian noise to the embeddings, with the scale decreasing over time. And at each optimization step, the updated embedding is projected to the nearest embedding in the LM vocabulary.

Eventhough [Kha+22] show that naively mapping an effecive soft prompt to their nearest tokens significantly drops the performance, the decreasing noise schedule seems to help.

Optimization

$$\tilde{e}^i = \text{Proj}_E \left[\tilde{e}^{i-1} - \eta \nabla_{\tilde{e}} \mathcal{E}(\tilde{e}^{i-1}) + \sqrt{2\eta\beta_i} z \right] \tag{1}$$

where:

- $\mathcal{E}$ energy function: $\mathcal{E}(\tilde{e}_{i-1}) = -\log p_\theta(v(y) \mid \tilde{e}^{i-1}, x, t)$
- $z \sim \mathcal{N}(0, I_{|\tilde{e}|})$ gaussian noise
- $\beta_{\text{start}} > \beta_i > \beta_{\text{end}} \to 0$ variance of the noise following a geometric progression
- E embedding table
- $\text{Proj}_E(\tilde{e}) = \arg\min_{e_v \in E}(\|e_v - \tilde{e}\|^2)$
- v verbalizer (e.g. to classify between positive and negative we use the tokens corresponding to "positive" and "negative")

Likelihood

$$p_\theta(\tilde{e}_m \mid \tilde{e}_{0:m-1}) = \frac{\exp(h_{\theta,m-1} \cdot \tilde{e}_m)}{\sum_{e_v \in E} \exp(h_{\theta,m-1} \cdot e_v)} \tag{2}$$

where $h_{\theta,m-1}$ is the last hidden state at position $m-1$.

$\tilde{e}_m$ is in the embedding table (because of the projection) hence it's equivalent to computing the logits.

They seems to suppose that $W_E = W_U$.

$$\mathcal{E}(\tilde{e}_{0:M}) = -\lambda_{\text{task}} \log p_\theta(v(y)|\tilde{e}_{0:M}, x, t) - \lambda_{\text{fluency}} \log p_\theta(\tilde{e}_{0:M}) \tag{3}$$

where $\lambda_{\text{task}} + \lambda_{\text{fluency}} = 1$.

Experiments

Two sentiment analysis tasks: Amazon Polarity and SST-2, and one topic classification task: AGNEWS. They choose these ones because vanilla soft prompting already improved perfomance on it but they did not try on RTE for example because vanilla soft prompting did not improve anything on it $\rightarrow$ maybe something to do with this.

Results: they see that FLUENTPROMPT performs comparably to AutoPromptSGD and significantly better than the empty prompt. In terms of readability, FLUENTPROMPT generates more fluent prompts than AutoPromptSGD. The fluency constraint effectively leads to significantly lower perplexity and also better accuracy.

Calibration

LMs are biased towards label words that are common in its training. To measure this they use a neutral string d as an input and compute the entropy of the labels (should be high as no label information is carried in d):

$$\begin{aligned} H(y) &= \mathbb{E}_{y \in Y} \left[-\log p(y) \right] \\ &= -\sum_{y \in Y} p_\theta(v(y) \mid \tilde{e}, d, t) \log p_\theta(v(y) \mid \tilde{e}, d, t) \end{aligned}$$

They observe that label entropy exhibits significant positive correlations with the task accuracy.

Calibration

Hence they decide to optimize the prompt towards greater calibration, with an (negative) entropy loss:

$$\mathcal{L}_{\text{entropy}}(\tilde{e}) = \mathbb{E}_{y \in Y} \left[\log \mathbb{E}_{x \in X} \, p_\theta(v(y) \mid \tilde{e}, x, t) \right] \tag{4}$$

Intuitively the entropy loss encourages the prompt to help model generate more balanced predictions at a group level.

Domain

They also observe that effective prompts overall are more related to the task domain. Hence they extend the fluency loss like this:

$$\begin{aligned}\mathcal{L}_{\text{domain}}(\tilde{e}) = &- \log p_\theta(\tilde{e}_{0:M}) \\ &- \sum_i \log p_\theta(x_i \mid \tilde{e}, x_{<i}) \\ &- \sum_j \log p_\theta(t_j \mid \tilde{e}, x, t_{<j})\end{aligned}$$

Finallly the unsupervised energy function $\mathcal{E}(\tilde{e}_{0:M})$ is updated to:

$$\mathcal{E}(\tilde{e}_{0:M}) = -\lambda_{\text{calibration}}\mathcal{L}_{\text{entropy}}(\tilde{e}) - \lambda_{\text{domain}}\mathcal{L}_{\text{domain}}(\tilde{e}) \tag{5}$$

where $\lambda_{\text{calibration}} + \lambda_{\text{domain}} = 1$.

Link

The domain relevance corresponds to the idea of optimizing the likelihood of the optimized token knowing the user input context. For example while optimizing "!!! London is the capital of" to output "France", p("!!!"|"London is the capital of") will be maximized instead of p("!!!").

1. FluentPrompt

2. Comparison with our solution

Comparison

Let's compare FluentPrompt with our solution:

	Our solution	FluentPrompt
Optimization step	$\tilde{p}^i = \tilde{p}^{i-1} - \eta \nabla_{\tilde{p}} \mathcal{L}(\tilde{p}^{i-1})$	$\tilde{e}^i = \text{Proj}_E \left[\tilde{e}^{i-1} - \eta \nabla_{\tilde{e}} \mathcal{E}(\tilde{e}^{i-1}) + \sqrt{2\eta\beta_i} z \right]$
Loss	$\mathcal{L}(\tilde{p}) = \log p(y\|x, \tilde{p}) + \log p(\tilde{p}) + \frac{1}{k} \sum_{j=1}^{k} H(\text{softmax}(W_E^T \tilde{p}_j))$	$\mathcal{E}(\tilde{e}_{0:M}) = -\lambda_{\mathbf{c}} \mathcal{L}_{\mathbf{e}}(\tilde{e}) - \lambda_{\mathbf{d}} \mathcal{L}_{\mathbf{d}}(\tilde{e})$

References I

[Kha+22] Daniel Khashabi et al. *Prompt Waywardness: The Curious Case of Discretized Interpretation of Continuous Prompts*. 2022. arXiv: 2112.08348 [cs.CL].

[Shi+22] Weijia Shi et al. *Toward Human Readable Prompt Tuning: Kubrick's The Shining is a good movie, and a good prompt too?* 2022. arXiv: 2212.10539 [cs.CL].