モデル間転移へ向けた 複数 LLM の幾何構造に基づくソフトプロンプトチューニング

岸本裕　小尾賢生　小杉哲　船越孝太郎　奥村学
東京科学大学
{ky,smalltail}@lr.first.iir.isct.ac.jp
{kosugi,funakoshi,oku}@first.iir.isct.ac.jp

概要

ソフトプロンプトチューニングは事前学習済み言語モデルのパラメータを固定したまま，付加した少数の連続プロンプトのみを最適化することで下流タスクへ効率的に適応させる手法である．昨今，ソースモデルで最適化したソフトプロンプトをターゲットモデルに適用するソフトプロンプト転移により新たなモデルに対するプロンプトチューニングのコスト削減が試みられている．しかし，転移したプロンプトは依然としてターゲットモデル上での直接の最適化に比べて性能が劣り，改善の余地がある．そこで本研究では転移を前提としたソフトプロンプトを設計することで，プロンプト転移性能の向上を目指す．具体的にはソフトプロンプトチューニングを複数モデルで同時に行うとともに，モデル間で共通する語彙の埋め込みと各モデルのソフトプロンプトの相対的な幾何構造が一致するような最適化制約を導入する．実験の結果，提案手法は転移性能を改善しうる一方で，改善量はソースモデルとタスクの組み合わせに依存することが判明した．

1　はじめに

事前学習済み言語モデルは自然言語処理の幅広いタスクで高い性能を示し，様々な用途に活用されている．そのような汎用的な知識をもつ大規模モデルを下流タスクへ適応させる手段として，ファインチューニングが一般的に用いられているが，モデル規模の増加に伴い要する計算資源は日々増大している．この負担を緩和するため，モデル本体のパラメータを固定したまま埋め込み空間にベクトル形式の連続プロンプトを付加し，勾配法等で最適化するソフトプロンプトチューニング [1, 2, 3, 4, 5] が軽量な適応手法として用いられている．

さらに，各モデルに対して個別にプロンプトチューニングを行うコストを削減するため，昨今ではソースモデルで最適化したソフトプロンプトをターゲットモデルに適用するソフトプロンプト転移 [6, 7] が提案されている．しかしながら，先行研究で提案された転移手法の性能はターゲットモデル上で直接最適化した場合に及ばないことが多く，依然として改善の余地が残る．

そこで本研究では，転移を前提としたソフトプロンプトを設計し，既存のプロンプト転移に対し性能の向上を目指す．

2　関連研究

2.1　ソフトプロンプトチューニング

ソフトプロンプトチューニングは，パラメータを凍結した事前学習済み言語モデルに対し，連続ベクトルからなるソフトプロンプトを最適化することで，下流タスクへ適応させる手法である [2, 3]．ソフトプロンプトを $P = [\mathbf{p}_1, \ldots, \mathbf{p}_m]^\top \in \mathbb{R}^{m \times d}$ と表す．ここで，$\mathbf{p}_x$ $(x = 1, \ldots, m)$ はプロンプト埋め込みベクトル，m はプロンプトトークン長，d はモデルの埋め込み次元である．データセット $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ に対して，プロンプトチューニングの目的は，

$$P^* = \underset{P}{\operatorname{argmax}} \sum_{i=1}^{n} \log p(y_i \mid [P; x_i]) \tag{1}$$

に従い P^* を求めることである．分類タスクでは，モデル出力からラベル語彙に対応付けたクラス確率を定義し，クロスエントロピー損失を最小化するように P を誤差逆伝搬法で更新する．

2.2　ソフトプロンプト転移

ソフトプロンプト転移は，ソースモデルで最適化したソフトプロンプトを，ターゲットモデルの埋め

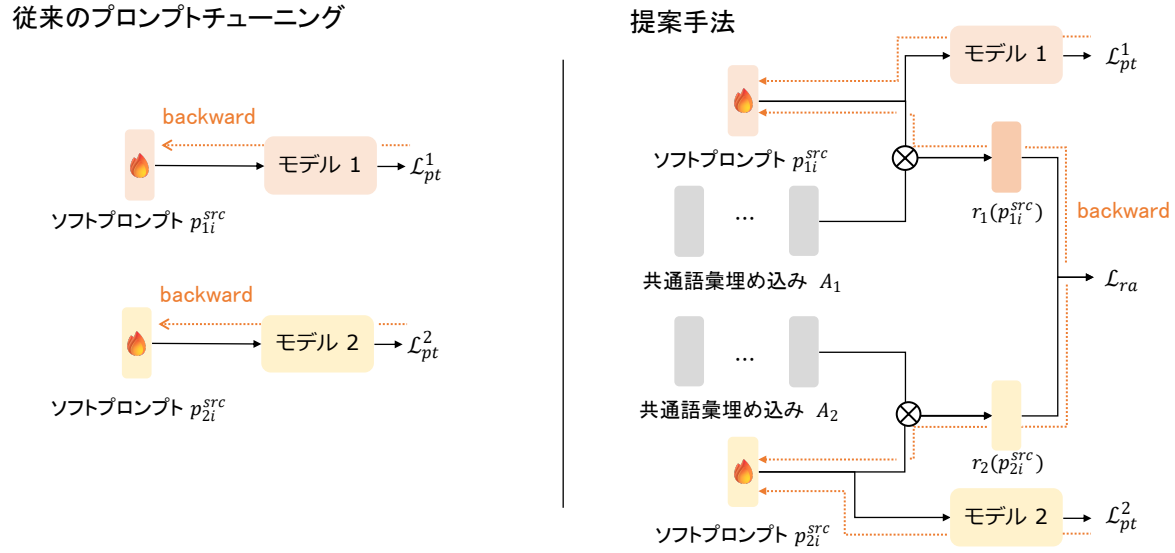

図 1: 提案手法のイメージ

通常のプロンプトチューニングはモデル毎に独立に実施するが，提案手法においては併せて共通語彙埋め込みとソフトプロンプトの相対的な幾何を整合させる損失 $\mathcal{L}_{\text{ra}}$ も同時に最適化する．

込み空間へ写像する手法である．ソースモデルで最適化されたプロンプトを $P^*_{\text{src}} \in \mathbb{R}^{m \times d_{\text{src}}}$，ターゲットモデルで最適化されたプロンプトを $P^*_{\text{tgt}} \in \mathbb{R}^{m \times d_{\text{tgt}}}$ とすると，ソフトプロンプト転移の目的は，P^*_{src} から写像 $f(\cdot)$ を介して，P^*_{tgt} の性能を近似する転移プロンプト $\tilde{P}_{\text{tgt}} \in \mathbb{R}^{m \times d_{\text{tgt}}}$ を構成することである．

$$\tilde{P}_{\text{tgt}} = f(P^*_{\text{src}}), \quad \tilde{P}_{\text{tgt}} \approx P^*_{\text{tgt}} \tag{2}$$

一般に，モデルごとに埋め込み空間の意味合いや次元が異なるため，単純な写像で転移すると性能が劣化しやすいことが知られている．

そこで Su ら [7] は，ソースモデルとターゲットモデルの埋め込み空間を媒介する変換器を学習し，ソースプロンプトをターゲット埋め込み空間へ写像する手法を提案した．しかしながら Su らの手法は変換器の学習にターゲットモデルの逆伝搬を必要としており要求される計算資源がターゲットモデルに応じて増大する．

そこで Wu ら [6] は，転移時にターゲットモデルの逆伝播が不要なゼロショット転移手法を提案した．彼らの手法はソース，ターゲットモデルの共通語彙を活用する．まず，ソースプロンプトとソースモデルの共通語彙埋め込みのコサイン類似度を計算し，これを相対表現と呼ぶ．ターゲット側も同じくランダムに初期化したプロンプトとターゲットモデルの共通語彙埋め込みのコサイン類似度を計算した相対表現を算出し，これら 2 つの相対表現同士の類似度を最大化するようなターゲットプロンプトを勾配法で探索する．これにより，転移時にターゲットモデルの逆伝播が不要なゼロショットプロンプト転移を実現した．さらに，複数ソースモデルの相対表現との整合性をターゲットモデルで同時に最大化することによりプロンプトのモデル依存性を緩和し単一ソース転移よりも性能が向上しうることを示した．本研究では，Wu らの転移手法について，単一ソースモデルのプロンプトから転移する場合をシングルソース転移，2 つのソースモデルのプロンプトから転移する場合をデュアルソース転移と呼ぶ．

3 提案手法

本研究では，Wu ら [6] のデュアルソース転移のアイデアに着想を得て，プロンプトチューニング時に複数モデル間で相対的な幾何構造が整合するような損失を設ける．これにより汎用的な幾何構造を有したプロンプトを最適化し，転移時のモデル間汎化性能向上を目指す．図 1 にイメージを示す．

S 個のソースモデルを $\{M_j\}_{j=1}^{S}$ とする．これらがもつ語彙をそれぞれ $\{\mathcal{V}_j\}$ とし，全てに共通する語彙を

$$\mathcal{V}_{\text{common}} = \bigcap_{j=1}^{S} \mathcal{V}_j \tag{3}$$

と定義する．ここで，共通語彙は 1 トークンのものに限定する．M_j における $\mathcal{V}_{\text{common}}$ に対応する語彙埋め込みを共通語彙埋め込み $A_j \in \mathbb{R}^{k \times d_j}$ と定義する．ここで k は共通語彙数である．各モデル M_j の

ソフトプロンプトを $P_j^{\mathrm{src}} = [\mathbf{p}_{j1}^{\mathrm{src}}; \ldots; \mathbf{p}_{jm}^{\mathrm{src}}]^\top \in \mathbb{R}^{m \times d_j}$ とし，$\mathbf{p}_{ji}^{\mathrm{src}}$ は j 番目のモデルの i 番目のプロンプトトークン埋め込みを示す．そして，$\mathbf{p}_{ji}^{\mathrm{src}}$ と A_j の相対表現を

$$r_j(\mathbf{p}_{ji}^{\mathrm{src}}) = c(\mathbf{p}_{ji}^{\mathrm{src}}, A_j) = \big(c(\mathbf{p}_{ji}^{\mathrm{src}}, \mathbf{a}_{j1}), \ldots, c(\mathbf{p}_{ji}^{\mathrm{src}}, \mathbf{a}_{jk})\big) \tag{4}$$

で定義する．ここで，$c(\cdot,\cdot)$ はコサイン類似度を表す．次に，モデル間で相対表現を整合させる損失を

$$\mathcal{L}_{\mathrm{ra}} = \frac{2}{S(S-1)} \sum_{1 \le j < j' \le S} \frac{1}{m} \sum_{i=1}^{m} \big(1 - c\big(r_j(\mathbf{p}_{ji}^{\mathrm{src}}), r_{j'}(\mathbf{p}_{j'i}^{\mathrm{src}})\big)\big) \tag{5}$$

と定義する．また，各モデル M_j における分類タスクの損失を $\mathcal{L}_{\mathrm{pt}}^j$ とし，その平均を

$$\mathcal{L}_{\mathrm{pt}} = \frac{1}{S} \sum_{j=1}^{S} \mathcal{L}_{\mathrm{pt}}^j \tag{6}$$

とおく．提案手法では，$\mathcal{L}_{\mathrm{pt}}$ と $\mathcal{L}_{\mathrm{ra}}$ を合わせた損失

$$\mathcal{L} = (1-\lambda)\mathcal{L}_{\mathrm{pt}} + \lambda \mathcal{L}_{\mathrm{ra}} \tag{7}$$

または

$$\mathcal{L} = \mathcal{L}_{\mathrm{pt}} + \eta_{\mathrm{annealing}} \mathcal{L}_{\mathrm{ra}} \tag{8}$$

を最適化する．ここで，λ は 2 つの損失の重みを調整する定数，$\eta_{\mathrm{annealing}}$ は学習の進行に応じて相対整合損失の重みを増加させる変数であり，

$$\eta_{\mathrm{annealing}} = \frac{1}{2}\left\{1 - \cos\left(\pi \frac{t}{T}\right)\right\} \tag{9}$$

で定義される．ここで t は現在の学習ステップ数，T は総学習ステップ数である．このとき，$\eta_{\mathrm{annealing}}$ は 0 から 1 へ滑らかに増加するため，学習初期は主に $\mathcal{L}_{\mathrm{pt}}$ が最適化され，学習が進むにつれて $\mathcal{L}_{\mathrm{ra}}$ の寄与が強まるような最適化が成される．

4 実験設定

ソースモデルには，BERT-base [8]，GPT-2 [9]，MPNet-base [10]，OPT-125M [11]，RoBERTa-base [12] を用いた．ターゲットモデルには RoBERTa-large [12] を用いた．

データセットには感情分類の Rotten Tomatoes movie reviews [13]，Sentiment Analysis [14, 15]，SST-2 [16, 17]，パラフレーズ分類の MRPC [17, 18]，皮肉検出の Irony Detection [19, 15]，ヘイトスピーチ検出の Hate Speech Detection [20, 15]，文章の攻撃性判定の Offensive Language Identification [21, 15] を用いた．検証データまたはテストデータのいずれかが提供されていない場合は，提供されている方をテストデータとして扱い，訓練データの 10%を検証データとして使用した．

本実験で用いるプロンプトトークン数は Wu ら [6] に倣い 5 トークンとした．また，本手法における最適化後のプロンプト転移には，Wu ら [6] の手法を用いてデュアルソース転移を用いた．プロンプトチューニングや転移に関するその他の情報は付録 A に示す．

5 結果

紙面の制約により，本文では BERT-base と他モデルをソースとする場合を報告する．その他の設定における実験の一部は付録 B に示す．また，提案手法の重み係数は $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$，$\eta_{\mathrm{annealing}}$ から検証データで選択し，良好であった設定の結果を示す．実験は異なるシード値を用いて 5 回行い，その平均値と標準偏差を報告する．表 1 にソースモデルのプロンプトチューニング性能を示す．表 2 には，プロンプトなしの場合のターゲットモデルの性能（Empty），手動のタスクテンプレートを付与した場合の性能（Manual），単一ソースモデルで最適化したプロンプトをシングルソース転移した性能（Single）を示す．最後に，表 3 に独立に最適化した 2 つのソースプロンプト（Baseline）と提案手法で最適化した 2 つのソースプロンプト（Ours）のデュアルソース転移の比較を示す．なお，プロンプトを含む入力の詳細は付録 C に示す．

まず表 1，表 3 より，Baseline，Ours ともに転移後プロンプトの性能はソースモデルにおけるプロンプトチューニング性能には及んでいない場合が多いことがわかる．

次に表 2 より，単一ソース転移（Single）は多くのタスクで Manual を上回っており，転移そのものが一定の有効性をもつことが確認できる．

最後に表 3 では，BERT-base と MPNet-base をソースとする組合せでは，提案手法（annealing）が Baseline を 7 タスク中 5 タスクで上回り，特に Hate と Irony で改善が見られた．一方で，BERT-base と RoBERTa-base の組合せでは，Rotten Tomatoes を中心に性能低下が確認された．また，Irony および MRPC では Baseline，Ours のすべての手法が Empty の性能を下回っている．

表 1: ソースモデルにおけるプロンプトチューニング性能

Models \ Tasks	Hate (M-F1)	Irony ($F1_{i=\text{irony}}$)	MRPC ($\frac{\text{Acc+F1}}{2}$)	Offensive (M-F1)	Rotten Tomatoes (Acc)	Sentiment (M-Rec)	SST-2 (Acc)
BERT-base	41.5 ± 1.8	59.3 ± 1.3	76.9 ± 1.5	73.4 ± 0.6	82.4 ± 0.4	66.3 ± 0.8	87.1 ± 0.4
GPT-2	48.9 ± 0.4	57.0 ± 0.1	74.5 ± 0.5	54.6 ± 2.7	80.6 ± 6.2	64.3 ± 0.9	86.0 ± 0.2
MPNET-base	41.3 ± 4.7	57.8 ± 1.3	76.1 ± 0.6	74.1 ± 0.8	87.7 ± 0.3	61.1 ± 2.4	91.5 ± 0.1
OPT-125M	42.9 ± 2.9	58.6 ± 0.9	75.7 ± 0.9	69.8 ± 5.1	85.2 ± 0.3	72.0 ± 0.3	89.9 ± 0.5
RoBERTa-base	40.4 ± 2.7	56.5 ± 2.9	76.3 ± 0.9	73.1 ± 0.9	86.4 ± 0.3	69.7 ± 0.8	91.9 ± 0.4

表 2: プロンプトチューニングなし / 単一モデルからの転移性能

		Target Model : RoBERTa-large						
		Tasks						
Src Prompt Type	Src Models	Hate (M-F1)	Irony ($F1_{i=\text{irony}}$)	MRPC ($\frac{\text{Acc+F1}}{2}$)	Offensive (M-F1)	Rotten Tomatoes (Acc)	Sentiment (M-Rec)	SST-2 (Acc)
Empty	-	32.3	65.9	74.8	40.5	67.8	44.3	71.0
Manual	-	35.4	58.2	74.8	39.8	72.6	46.3	76.7
Single	BERT-base	32.6 ± 0.8	62.5 ± 0.9	73.9 ± 0.6	42.1 ± 0.9	77.6 ± 0.7	49.1 ± 0.2	78.9 ± 1.0
	GPT-2	31.3 ± 0.2	61.8 ± 1.4	73.9 ± 0.8	42.3 ± 0.6	76.3 ± 0.8	49.5 ± 0.2	80.4 ± 0.5
	MPNet-base	32.5 ± 0.3	61.0 ± 0.6	74.3 ± 0.5	40.7 ± 0.2	61.1 ± 1.5	46.4 ± 0.9	80.8 ± 0.3
	OPT-125M	32.4 ± 0.2	60.9 ± 1.3	74.5 ± 0.4	40.5 ± 0.5	61.3 ± 0.4	44.4 ± 0.5	78.0 ± 2.6
	RoBERTa-base	33.0 ± 2.0	62.3 ± 1.7	73.0 ± 0.9	42.1 ± 0.7	77.6 ± 1.1	49.2 ± 0.6	77.6 ± 1.8

表 3: 従来手法，提案手法の性能比較

	Target Model : RoBERTa-large						
	Tasks						
Src Prompt Type	Hate (M-F1)	Irony ($F1_{i=\text{irony}}$)	MRPC ($\frac{\text{Acc+F1}}{2}$)	Offensive (M-F1)	Rotten Tomatoes (Acc)	Sentiment (M-Rec)	SST-2 (Acc)
BERT-base + GPT-2							
Baseline	**31.7 ± 0.3**	62.3 ± 1.2	74.1 ± 0.6	42.4 ± 0.5	**77.0 ± 0.4**	49.3 ± 0.2	**79.3 ± 1.1**
Ours λ=0.3	31.5 ± 0.2	**63.2 ± 1.2**	**74.4 ± 0.5**	**42.8 ± 0.7**	75.8 ± 0.7	**49.4 ± 0.3**	79.2 ± 1.5
BERT-base + MPNet-base							
Baseline	32.9 ± 1.4	62.2 ± 2.4	74.4 ± 0.4	**42.4 ± 0.8**	78.0 ± 0.9	49.2 ± 0.3	81.5 ± 0.4
Ours $\eta_{\text{annealing}}$	**33.8 ± 0.4**	**63.3 ± 1.6**	**74.5 ± 0.6**	42.1 ± 0.5	**78.3 ± 0.8**	49.1 ± 0.3	**81.8 ± 1.0**
BERT-base + OPT-125M							
Baseline	**33.6 ± 0.5**	61.0 ± 1.4	**74.2 ± 0.7**	42.4 ± 0.6	77.0 ± 0.7	**48.5 ± 0.7**	**78.7 ± 1.3**
Ours λ=0.3	33.4 ± 0.4	**61.7 ± 1.3**	**74.2 ± 0.5**	**42.8 ± 0.7**	**77.3 ± 0.6**	48.1 ± 0.5	**78.7 ± 2.4**
BERT-base + RoBERTa-base							
Baseline	**33.1 ± 1.2**	62.1 ± 0.7	**74.0 ± 0.8**	**42.1 ± 0.7**	**78.4 ± 1.9**	**49.5 ± 0.9**	79.4 ± 1.9
Ours λ=0.1	32.6 ± 1.0	**63.4 ± 1.3**	73.7 ± 0.6	41.3 ± 1.1	73.5 ± 1.6	49.2 ± 0.2	**80.6 ± 2.5**

5.1 結果の解釈

転移プロンプトはソースモデルのプロンプトチューニング性能に及ばない場合が殆どであり，プロンプト転移による性能低下の回避は困難であることがわかる．

デュアルソース転移に関して，提案手法で最適化したプロンプトの転移性能はソースモデルの組み合わせやタスクに強く依存することがわかる．特に RoBERTa-base はターゲットである RoBERTa-large と同系統であるため，提案手法の損失により不要な整合が行われ，転移に不利に働いた可能性がある．ただし，この点は追加分析により検証が必要である．

また，一部タスクでは Empty を下回るなど，転移に有効なプロンプトの最適化自体が難しいことが示唆され，より強力なプロンプトチューニング手法を適用するなどの余地がある．

6 おわりに

本研究では，ソフトプロンプトチューニングにおいてソフトプロンプトと共通語彙埋め込みに基づく相対表現を複数ソースモデル間で整合させる損失を導入し，転移性能への影響を評価した．その結果，一部のモデルの組合せとタスクにおいて転移性能を改善しうることを確認した一方で，改善幅はモデルの組合せとタスクに強く依存し，常に一貫した改善は得られなかった．転移に有効なソースモデルの選択条件と，性能変動の要因の解明は今後の課題である．

参考文献

[1] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 4222–4235, Online, November 2020. Association for Computational Linguistics.

[2] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In **Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing**, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[3] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: Learning vs. learning to recall. In **Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 5017–5033, Online, June 2021. Association for Computational Linguistics.

[4] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric Xing, and Zhiting Hu. RLPrompt: Optimizing discrete text prompts with reinforcement learning. In **Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing**, pp. 3369–3391, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[5] Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. In **Thirty-seventh Conference on Neural Information Processing Systems**, 2023.

[6] Zijun Wu, Yongkang Wu, and Lili Mou. Zero-shot continuous prompt transfer: Generalizing task semantics across language models. In **The Twelfth International Conference on Learning Representations**, 2024.

[7] Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. On transferability of prompt tuning for natural language processing. In **Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 3949–3969, Seattle, United States, July 2022. Association for Computational Linguistics.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[10] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. In **Advances in Neural Information Processing Systems**, Vol. 33, pp. 16857–16867. Curran Associates, Inc., 2020.

[11] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

[12] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Ro{bert}a: A robustly optimized {bert} pretraining approach, 2020.

[13] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In **Proceedings of the ACL**, 2005.

[14] Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. In **Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)**, pp. 502–518, 2017.

[15] Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In **Proceedings of Findings of EMNLP**, 2020.

[16] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In **Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing**, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[17] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. 2019. In the Proceedings of ICLR.

[18] William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In **Proceedings of the International Workshop on Paraphrasing**, 2005.

[19] Cynthia Van Hee, Els Lefever, and Véronique Hoste. Semeval-2018 task 3: Irony detection in english tweets. In **Proceedings of The 12th International Workshop on Semantic Evaluation**, pp. 39–50, 2018.

[20] Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In **Proceedings of the 13th International Workshop on Semantic Evaluation**, pp. 54–63, Minneapolis, Minnesota, USA, 2019. Association for Computational Linguistics.

[21] Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In **Proceedings of the 13th International Workshop on Semantic Evaluation**, pp. 75–86, 2019.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. **CoRR**, Vol. abs/1412.6980, , 2014.

表 4: RoBERTa-base とその他のモデルをソースモデルとした際の転移結果

Target Model : RoBERTa-large							
	Tasks						
Src Prompt Type	Hate (M-F1)	Irony ($F1_{i=\text{irony}}$)	MRPC ($\frac{\text{Acc+F1}}{2}$)	Offensive (M-F1)	Rotten Tomatoes (Acc)	Sentiment (M-Rec)	SST-2 (Acc)
GPT-2 + RoBERTa-base							
Baseline	**35.1 ± 3.4**	62.9 ± 1.0	73.7 ± 0.2	41.9 ± 0.2	**77.0 ± 3.7**	49.3 ± 0.3	80.5 ± 0.6
Ours $\lambda_{\cos}$=0.5	32.6 ± 1.3	**63.4 ± 1.4**	73.7 ± 0.4	**42.1 ± 0.6**	75.5 ± 1.1	**49.7 ± 0.2**	**81.0 ± 0.9**
MPNet-base + RoBERTa-base							
Baseline	**34.0 ± 3.0**	60.9 ± 1.2	74.0 ± 0.5	**42.4 ± 0.7**	**78.9 ± 1.2**	**49.7 ± 0.2**	79.2 ± 1.7
Ours $\eta_{\text{annealing}}$	31.6 ± 0.4	**61.9 ± 1.2**	**74.2 ± 0.6**	41.8 ± 0.8	77.3 ± 1.2	49.2 ± 0.4	**80.5 ± 1.6**
OPT-125M + RoBERTa-base							
Baseline	**36.5 ± 2.7**	**63.3 ± 0.7**	73.0 ± 1.8	**42.0 ± 1.0**	76.9 ± 0.7	49.2 ± 0.5	79.4 ± 1.6
Ours $\eta_{\text{annealing}}$	32.0 ± 0.7	61.3 ± 1.0	**73.7 ± 0.8**	41.7 ± 0.9	**77.5 ± 0.9**	**49.4 ± 0.3**	**79.6 ± 0.8**

A 実験パラメータ

プロンプトチューニングのパラメータに関しては Wu ら [6] に従い，OptiPrompt [3] に準拠している．プロンプト転移に関して，表 5 にプロンプトチューニングのパラメータを示す．基本的に Wu ら [6] に従うが，実験タスクの収束性を鑑みて学習ステップを 2000 → 10000 へ増加させている．

表 5: プロンプトチューニングのパラメータ

Parameter	Value
Learning rate	5e-3
Training steps	10000
Optimizer	Adam [22]
The number of anchors	8192

B 追加の実験結果

表 6 に RoBERTa-base とその他のモデルをソースモデルとした際の転移結果を示す．表 3 の BERT-base RoBERTa-base がソースモデルの場合と併せて，SST-2 においては良好な性能を示す一方 Hate では大きく性能が劣化することが確認できる．従って本文で述べたとおり，本手法はモデル，タスクの組み合わせにより性能差が異なり，最適なモデル，タスクの組み合わせを解明することは今後の課題である．

C 入力文の形式

表 6 に本研究で使用した入力文章の形式を示す．OptiPrompt [3] に従い入力文を構成した．$[X]_{n=1}^{m}$ はソフトプロンプト m トークン，$[Y]$ は MASK トークンを表す．また，$\langle s \rangle$ はデータセット文章，

表 6: 入力文

Method	Prompt
Empty	$\langle s \rangle$ $[Y]$
Manual	$[X]_{n=1}^{m}$ $\langle s \rangle$ $\langle template \rangle$ $[Y]$
Prompt Tuning	$[X]_{n=1}^{m}$ $\langle s \rangle$ $[Y]$

表 7: テンプレート

Dataset	$\langle template \rangle$
Rotten Tomatoes movie reviews	It was
Sentiment Analysis	It was
SST-2	It was
MRPC	Are the sentences equivalent?
Irony Detection	It was
Hate Speech Detection	It was
Offensive Language Identification	It was

$\langle template \rangle$ は Manual の際の簡単なテンプレート，なお，MASKed 言語モデル以外は $[Y]$ は存在せず，入力文の直後のトークンを直接予測させる．また，表 7 に Manual の際に使用したテンプレートを示す．

D 開示事項

本研究の着想検討および構成の整理にあたり，対話型生成 AI（ChatGPT）を補助的に利用した．