タスク算術の誤差項とその解釈

李 宰成 [1]　中石 海 [2,3]　鈴木 潤 [1,2,4]　横井 祥 [3,1,2]
[1] 東北大学　[2] 理化学研究所　[3] 国立国語研究所　[4] 国立情報学研究所 LLMC
lee.jaesung@dc.tohoku.ac.jp, kai.nakaishi@riken.jp
yokoi@ninjal.ac.jp

概要

重み空間での「タスク算術」によって，追加学習抜きに言語モデルに新しいスキルを付与できる可能性がある．例えば，特定のタスクへの微調整前後のモデル重みの差分を「タスクベクトル」として取り出し，これを複数足し合わせることで，複数タスクに対応したモデルを構築できる．ここで，個々のタスクそれぞれに適応したモデルを作る場合は成功例が多い一方，「要約して翻訳する」のような構成的なタスクではこれが困難なことが経験的に知られている．本稿では，単語埋め込みの加法構成性の理論に着想を得て，タスクベクトルの加法構成性の定式化を試みる．提案する理論的枠組みにより，タスク算術の成否に関わる近似誤差を導出できる．またこの帰結として，複数タスクのそれぞれを実現しようとする「マルチタスク学習」ではこの近似誤差が消える一方，構成的なタスクを含む一般的なタスクでは多数の誤差項が導かれることを示した．

1　はじめに

事前学習済み言語モデルを特定のタスクに適応させるもっとも標準的な方法は，その微調整であろう．しかしモデルの規模は増大の一途をたどっており，必要なスキルの組み合わせに応じて次々と微調整を実施するのは，計算資源を持たぬユーザーにとって最早容易な方法ではなくなっている．

最近発見されたタスク算術（task arithmetic）[1] は，数回の微調整をおこなうだけで，あるいはインターネット上から微調整済みモデルをダウンロードするだけで，あとはモデル重みの足し引きによって新しいモデルを作ることのできる技術である [2, 3, 4, 5]．より具体的には，特定のタスクでの微調整前後のモデル重みの差分をタスクベクトルと呼び，事前学習済みモデルに対し，好きな組み合わせでタスクベクトルたちを線形に足し合わせることで，複数のタスクに適応したモデルを構築できる．ただしタスク算術は常にうまくいくわけではなく，複数のタスクをそれぞれに実行するモデルは作成しやすい一方，「要約して翻訳する」のような複数のタスクを組み合わせた構成的タスクを実現するモデルの作成は困難であると経験的に知られている [6]．

本稿では，単語埋め込みの加法構成性の理論に着想を得て，**タスクベクトルの加法構成性**の理論の構築を試みる．特に，各タスクの損失に議論を帰着させながら，複数のタスクベクトルの線形和と，目標とするタスクベクトルの関係を定式化する．

理論的帰結として，複数タスクをそれぞれ実現するような新しいモデルを作ろうとする場合（マルチタスク学習）は加法構成の近似誤差が生じず，一方で構成的なタスクを含む一般的なタスクでは多数の誤差項が導かれることがわかった．すなわち現実のタスク算術の成否についてある程度説明可能な，ひとつの枠組みを構築できたと考える．本研究を足掛かりに，まだ不明点の多いタスク算術の各種性質の解明が進むことを期待したい．

2　背景：タスク算術

タスクベクトル　**タスクベクトル**とは，タスクの微調整によるモデルの重みの変化をベクトルで表したものである．すなわち，事前学習済みモデルの重みを $\boldsymbol{\theta}_0 \in \mathbb{R}^d$，微調整後の重みを $\boldsymbol{\theta} \in \mathbb{R}^d$ として，タスクベクトルは $\boldsymbol{\tau} = \boldsymbol{\theta} - \boldsymbol{\theta}_0 \in \mathbb{R}^d$ で定義される．なお，$\boldsymbol{\theta_0}$ と $\boldsymbol{\theta}$ はモデル全体の重みに限らず，LoRA [7] により学習される低ランク行列としてみなしてよいとする[1)]．

タスクベクトルは重み空間におけるタスクの表現であると捉えられる．**タスク算術** [1] は，この考え方に従い，複数のタスクベクトルを組み合わせて

1)　LoRA で導入される低ランク行列の学習前後の差分ベクトルを，線形和および線形結合することで，タスク算術と同様の効果が得られることが経験的に報告されている [8]．

新たなモデルを構築する手法である．具体的には，K 個のタスク $1,\ldots,K$ に対応するタスクベクトル $\boldsymbol{\tau}_1,\cdots,\boldsymbol{\tau}_K$ の線形和を構成し，事前学習済みモデルの重みに加算することで，新たなモデルの重み

$$\boldsymbol{\theta}_{\text{new}} = \boldsymbol{\theta}_0 + \sum_{i=1}^{K} \lambda_i \boldsymbol{\tau}_i. \tag{1}$$

を得る．係数 $\lambda_1,\cdots,\lambda_K$ は，新たなモデルが目指す性能を達成するよう，探索により決められる．

マルチタスク学習への適用　タスク算術により実現したいモデルとして，複数のタスクで同時に微調整した場合と同等の性能を持つモデルが挙げられる．例えば，文章を要約するタスクと文章をフォーマルな文体に言い換えるタスクの双方を高い水準で実行するモデルなどである．本稿では，複数のタスクによる単一のモデルの微調整を**マルチタスク**学習と呼ぶ．

タスク算術はマルチタスク学習に近い効果を実現できることが経験的に知られている [1, 4]．理論的研究も進められており [9, 10]，特に，ある仮定のもとでは，マルチタスク学習で得られるタスクベクトルをタスク算術で構成できることが示されている [11]．

構成的タスクへの適用　別の目標として，タスク算術によって，もともとのタスクとは異なる新たなタスクで微調整した場合と同等の効果を得ることも考えられる．例えば，英文を日本語に翻訳するタスクと，英文を英語で要約するタスクから得たタスクベクトルを組み合わせて，英文を日本語で要約するモデルの構成を試みる場合である．以下，このようなもともとのタスクを組み合わせたものとして捉えられる新たなタスクを**構成的タスク**と呼ぶ．

先行研究の実験からは，タスク算術でこのような目標を達成するのは困難だと示唆されている．例えば，線形マージと呼ばれる手法 [12] をタスク算術と等価とみなせる条件下で[2)]構成的タスクに適用した研究では，期待した性能向上は得られなかった [13]．同様に，タスク算術の改良手法 [3, 4] を構成的タスクへ適用した場合にも，十分な性能は得られなかった [6]．

ここから，タスク算術で実現できる目標にはなんらかの限界があることが示唆される．しかし，ほとんどのタスク算術の理論研究がマルチタスク学習の場合に限られており，構成的タスクを含むより一般の場合における有効性は明らかにされていない．

単語埋め込みの加法構成性　この問題にアプローチするべく，本稿では，単語埋め込みの加法構成性の考え方を援用する．ある単語の単語埋め込みは，それを言い換える複数の単語の単語埋め込みの線形和で近似でき，この性質は**加法構成性**と呼ばれる [14]．この性質は，単語埋め込みが対照学習により得られる状況では，ある単語埋め込みとその言い換えに対応する単語埋め込みの線形和との差を評価することで，理論的に説明できる [15]．

本稿では，タスクベクトルの線形和によって目標とするタスクベクトルを近似できることをタスクベクトルの加法構成性と呼ぶ．そして，各タスクベクトルが適当な損失のもとでの学習によって得られるという状況のもと，目標とするタスクベクトルと要素となるタスクベクトルの線形和の差を，各タスクの損失を含んだ形で導出する．

3　タスク算術の誤差の導出と解釈

タスクベクトルの加法構成性を導入し，この性質がどれだけ破れているかを表す誤差を導く．なお，以降の議論は，タスクベクトルを LoRA で追加する低ランク行列の学習前後の差分として捉えた場合にも同様に成り立つ．

3.1　加法構成性

タスクベクトルの加法構成性を導入する．K 個のタスクに対応するタスクベクトル $\boldsymbol{\tau}_1,\cdots,\boldsymbol{\tau}_K$ の線形和 $\sum_{i=1}^{K}\lambda_i\boldsymbol{\tau}_i$ によって，目標とするタスク $*$ のタスクベクトル $\boldsymbol{\tau}_*$ を近似することを考える．この近似誤差を $\boldsymbol{\Delta} \coloneqq \boldsymbol{\tau}_* - \sum_{i=1}^{K}\lambda_i\boldsymbol{\tau}_i$ とおく．

本稿では，ある係数の集合 $\lambda_1,\cdots,\lambda_K$ が存在して，近似誤差 Δ が十分小さくなるとき，タスクベクトルの**加法構成性**が成り立つと言う．加法構成性が成り立つならば，タスク $*$ による微調整と同等の効果をタスク算術によって得られる．

3.2　タスクベクトル間の関係

タスクの損失から手発し，タスクベクトル間の関係を導く．簡単のため，要素となるタスクの数が $K=2$ の場合を考える．タスク $i\in\{1,2,*\}$ に対応する損失を $L_i(\boldsymbol{\theta})$ とする．$L_1(\boldsymbol{\theta})$ と $L_2(\boldsymbol{\theta})$ の線形和と $L_*(\boldsymbol{\theta})$ の差分を $\mathscr{E}(\boldsymbol{\theta}) = L_*(\boldsymbol{\theta}) - \alpha_1 L_1(\boldsymbol{\theta}) - \alpha_2 L_2(\boldsymbol{\theta})$ とおく．$\alpha_1,\alpha_2\in\mathbb{R}$ は任意のスカラー値である．する

2)　線形マージ [12] は，同じ事前学習済みモデルから微調整された複数の派生モデルに適用する場合，タスクベクトルの線形和と等価な操作として解釈できる．

と，タスク $*$ の損失は以下のように表せる：

$$L_*(\boldsymbol{\theta}) = \alpha_1 L_1(\boldsymbol{\theta}) + \alpha_2 L_2(\boldsymbol{\theta}) + \mathscr{C}(\boldsymbol{\theta}). \tag{2}$$

これを $\boldsymbol{\theta}_0$ 周りで微分し，以下が得られる：

$$\nabla L_*(\boldsymbol{\theta}_0) = \alpha_1 \nabla L_1(\boldsymbol{\theta}_0) + \alpha_2 \nabla L_2(\boldsymbol{\theta}_0) + \nabla \mathscr{C}(\boldsymbol{\theta}_0). \tag{3}$$

ここで，各タスクに対応するタスクベクトルが，$\boldsymbol{\theta}_0$ での勾配のスカラー倍によって近似されると仮定する．すなわち，$\boldsymbol{\tau}_i \approx \eta_i \nabla L_i(\boldsymbol{\theta}_0)$ とする．これは，微調整による重みの更新の方向が $\nabla L_i(\boldsymbol{\theta}_0)$ と似た方向を向いていることを意味する．例えば，先行研究 [11] では，微調整がフルバッチ学習を勾配降下法のもと 1 エポックのみ行われる状況を仮定しており，この場合には上記の近似は自然に成り立つ．また，学習が複数エポックに渡る場合にも，各エポックでの更新が平均的に $\nabla L_i(\boldsymbol{\theta}_0)$ と同じ方向を向いていれば，この仮定は成り立つ．

この仮定と式 (3) から，

$$\boldsymbol{\tau}_* \approx \frac{\alpha_1 \eta_*}{\eta_1} \boldsymbol{\tau}_1 + \frac{\alpha_2 \eta_*}{\eta_2} \boldsymbol{\tau}_2 \; - \; \eta_* \nabla \mathscr{C}(\boldsymbol{\theta}_0) \tag{4}$$

が導かれる．したがって，$\nabla \mathscr{C}(\boldsymbol{\theta}_0)$ が十分小さくなるような α_1, α_2 が存在すれば，近似誤差 Δ が十分小さくなり，加法構成性が成り立つ．

以上より，加法構成性の成立条件は，$\nabla \mathscr{C}(\boldsymbol{\theta}_0)$ に強く依存している．以下では．目標とするタスクがマルチタスクである場合と一般的なタスクである場合のそれぞれにおいて，この項を導出する．

3.3 マルチタスク

まず，タスク算術によってタスク 1 と 2 のマルチタスク学習と同等の効果を得たい場合を考える．マルチタスク学習においては，適当な係数 α_1', α_2' をとり，タスク 1 と 2 の損失の線形和 $\alpha_1' L_1(\boldsymbol{\theta}) + \alpha_2' L_2(\boldsymbol{\theta})$ を損失として微調整をおこなえば良いと期待される．よって，目標となるタスクベクトルに対応する損失は $L_* = \alpha_1' L_1(\boldsymbol{\theta}) + \alpha_2' L_2(\boldsymbol{\theta})$ である．

このとき，α_1 と α_2 を $\alpha_1 = \alpha_1'$ および $\alpha_2 = \alpha_2'$ と選べば，任意の θ において $\mathscr{C}(\boldsymbol{\theta}) = 0$ であり，$\nabla \mathscr{C}(\theta_0) = 0$ である．よって加法構成性が成り立つ．

このように，マルチタスク学習に対応するタスクベクトルは，タスク算術によって構成することができる．この結論は，タスク算術によってマルチタスク学習と同等の効果が得られることを示す先行研究の結果と整合的である．

以上の議論は，タスクが K 個ある場合においても同様に成り立つ．

3.4 一般的なタスク

次に，タスク $*$ が構成的タスクを含む一般的なタスクである場合を考える．

3.4.1 誤差項の分解

より具体的に考えるため，各タスクによる微調整がクロスエントロピーにもとづいておこなわれる状況を考える．タスク i のデータ分布を q_i とし，入力 x と出力 y のペアがこの分布から生成されるとする．また，タスクを解く際にモデルに与える指示文を I_i とする．さらに，重み $\boldsymbol{\theta}$ のモデルの出力分布を $p_{\boldsymbol{\theta}}$ とする．この状況では，このタスクの損失は $L_i(\boldsymbol{\theta}) = -\mathop{\mathbb{E}}_{q_i} [\log p_{\boldsymbol{\theta}}(y|I_i, x)]$ である．

このとき，$\nabla \mathscr{C}(\boldsymbol{\theta}_0)$ は以下のように分解される：

$$\nabla \mathscr{C}(\boldsymbol{\theta}_0) = \boldsymbol{\delta}_1 + \boldsymbol{\delta}_2 + \boldsymbol{\delta}_*, \tag{5}$$

$$\boldsymbol{\delta}_1 = -\sum_{x,y} (q_*(x,y) - \alpha_1 q_1(x,y)) \nabla \log p_{\boldsymbol{\theta}_0}(y|I_1, x), \tag{6}$$

$$\boldsymbol{\delta}_2 = -\sum_{x,y} (q_*(x,y) - \alpha_2 q_2(x,y)) \nabla \log p_{\boldsymbol{\theta}_0}(y|I_2, x), \tag{7}$$

$$\boldsymbol{\delta}_* = -\mathop{\mathbb{E}}_{q_*} \left[\nabla \log \frac{p_{\boldsymbol{\theta}_0}(y|I_*, x)}{p_{\boldsymbol{\theta}_0}(y|I_1, x) p_{\boldsymbol{\theta}_0}(y|I_2, x)} \right]. \tag{8}$$

導出過程は付録 A.1 に示す．以下では，$\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \boldsymbol{\delta}_*$ の各項について順に考察する．

3.4.2 データ分布の不一致に由来する項

項 $\boldsymbol{\delta}_1$ は，ある α_1 が存在して，タスク 1 のデータ分布の α_1 倍 $\alpha_1 q_1$ によりタスク $*$ のデータ分布 q_* をよく近似できる場合に，小さくなる．このことから，この項はタスク 1 とタスク $*$ のあいだでのデータ分布の不一致に由来すると解釈できる．

ただし，タスク 1 のデータ点ごとの損失の勾配 $\nabla \log p_{\boldsymbol{\theta}_0}(y|I_1, x)$ との積がとられていることに注意すべきである．つまり，あるデータ点において分布 q_1 と q_* が乖離していたとしても，そのデータ点における $\boldsymbol{\theta}_0$ でのタスク 1 の勾配 $\nabla \log p_{\boldsymbol{\theta}_0}(y|I_1, x)$ のノルムが小さければ，$\boldsymbol{\delta}_1$ への寄与は小さい．

同様に，項 $\boldsymbol{\delta}_2$ はタスク 2 とタスク $*$ のあいだでのデータ分布の不一致に由来すると解釈できる．

3.4.3 出力分布の不一致に由来する項

項 $\boldsymbol{\delta}_*$ 中の $\log \frac{p_{\boldsymbol{\theta}_0}(y|I_*, x)}{p_{\boldsymbol{\theta}_0}(y|I_1, x) p_{\boldsymbol{\theta}_0}(y|I_2, x)}$ は，事前学習済みモデルの指示文 I_* のもとでの出力分布 $p_{\boldsymbol{\theta}_0}(y|I_*, x)$ が，I_1 と I_2 のもとでの出力分布の積 $p_{\boldsymbol{\theta}_0}(y|I_1, x) p_{\boldsymbol{\theta}_0}(y|I_2, x)$ の定数倍で近似されるような

単純な分布から，どれだけ乖離しているかを表す．よって，項 $\boldsymbol{\delta}_*$ はこの乖離に由来するものと解釈される．ただし，$\log \frac{p_{\boldsymbol{\theta}_0}(y|I_*,x)}{p_{\boldsymbol{\theta}_0}(y|I_1,x)p_{\boldsymbol{\theta}_0}(y|I_2,x)}$ が大きいことは，必ずしもその勾配が大きいことを意味しない点は注意を要する．

また，この項はタスク $*$ のデータ分布 q_* のもとでの平均も含んでいる．従って，指示文 I_* のもとでの出力分布が，あるデータ点において指示文 I_1 と I_2 のもとでの出力分布の積から乖離していたとしても，そのデータ点がタスク $*$ においてほとんど生成されないのであれば，この項には寄与しない．

3.4.4 構成的タスク

ここまで，構成的タスクに限らない一般的なタスク $*$ について議論してきた．タスク $*$ が構成的タスクである場合，特に指示文 I_* がタスク 1 と 2 の指示文 I_1 と I_2 を連結したもので与えられる場合には，項 $\boldsymbol{\delta}_*$ をさらに分解できる．ここで，指示文 I_* が I_1 と I_2 の連結で与えられるとは，タスク $1, 2, *$ の指示文がそれぞれ「要約せよ」，「翻訳せよ」，「要約して翻訳せよ」であるような場合を意味している．

このとき，項 $\boldsymbol{\delta}_*$ は以下のように分解できる：

$$\begin{aligned}\boldsymbol{\delta}_* = &- \mathop{\mathbb{E}}_{q_*}\left[-\nabla \log p_{\boldsymbol{\theta}_0}(y|x)\right] \\ &- \mathop{\mathbb{E}}_{q_*}\left[\nabla\left(\mathrm{SI}(I_1;I_2;x) - \mathrm{SI}(I_1;I_2;x,y)\right)\right] \qquad (9)\end{aligned}$$

導出は付録 A.2 に示す．

ここで，SI は 3 変数に拡張された点ごとの相互情報量であり，$\mathrm{SI}(u;v;w) = \log \frac{p(v,w)}{p(v)p(w)} - \log \frac{p(v,w|u)}{p(v|u)p(w|u)}$ で定義される [16][3]．例えば，$\mathrm{SI}(I_1;I_2;x)$ は，指示文 I_2 と入力 x の系列 I_2, x の直前に指示文 I_1 が出現した際の影響の強さを表す．

式 (9) の右辺第 1 項に現れる $p_{\boldsymbol{\theta}_0}(y|x)$ は，モデルに指示文を与えず，x から y を予測する際の出力確率である．よって，この項は，指示文を与えない設定において，タスク $*$ のデータ分布 q_* のもと定義される損失の勾配に，負号を付したものである．

式 (9) の右辺第 2 項は，指示文 I_1, I_2 と入力 x および入出力 x, y の系列に関する点ごとの相互情報量 $SI(I_1;I_2;x)$，$SI(I_1;I_2;x,y)$ に由来する項である．例えば，I_2 と x の関連の強さが，指示文 I_1 が直前の文脈として現れる場合と現れない場合とで変化しないとき，$SI(I_1;I_2;x)$ は小さくなる．なお，SI が小さいとしても，その勾配が小さくなるとは限らない．

3) この量は相互作用情報量 [17] に基づくものであり，全相関 [18] に基づく素朴な点ごとの相互情報量とは区別される．

4 議論

マルチタスクの場合には，加法構成の誤差は生じない．これは，タスク算術によってマルチタスク学習と同等の効果が得られるという経験的知見と整合的である．一方，構成的タスクを含むより一般の場合には，データ分布やモデルの出力分布の不一致に由来する項などの様々な項がこの誤差に寄与する．これらの項が全て小さいか，互いに打ち消し合う状況でない限り，タスク算術は有効ではない．

タスク算術をより深く理解するには，目標とするタスクの特徴をより具体的に取り込みながら，誤差の各項を評価する必要があるだろう．例えば，本稿では構成的タスクの近似誤差を指示文に注目して導出したが，タスクのデータ分布の性質は考慮しなかった．この性質を考慮して誤差を評価することで，構成的タスクの困難さの要因を，より具体的に指摘できるだろう．

また，タスク算術が有効だと知られているマルチタスク以外の例として，タスクアナロジーによるドメイン転移がある [1, 19]．例えば，Amazon レビューの感情分析，Amazon レビューの次単語予測，Yelp レビューの次単語予測のタスクベクトルを $\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3$ として，$\lambda_1\boldsymbol{\tau}_1 - \lambda_2\boldsymbol{\tau}_2 + \lambda_3\boldsymbol{\tau}_3$ によって，Yelp レビューの感情分析をおこなうモデルを構築できる．本稿の枠組みは，要素となるタスクの数を 3 つ以上に拡張すれば，この問題にも適用可能である．ここでも，各タスクの指示文とデータ分布の特徴を考慮することは重要だろう．

今回の結果が実用的タスクにおけるタスク算術の振る舞いと整合するか，実験的に検証することも必要である．具体的には，本稿で導出した誤差の各項を実際に計測し，タスク算術の性能との相関を定量的に調べることが考えられる．

5 おわりに

本稿では，タスク算術の成否を理解・議論するための理論的枠組みとして，タスクベクトルの加法構成性を近似誤差項つきで定式化した．帰結として，タスク算術の目標がマルチタスク的である場合には誤差が生じないこと，目標が構成的なタスクを含むより一般的なケースでは複数の要因に由来する誤差項が生じることを示した．本稿の枠組みを発展させ，さらに経験的な検証と組み合わせることで，タスク算術についての精緻な理解を得ていきたい．

謝辞

本研究は，JSPS 科研費研究活動スタート支援 25K24434，JST 創発 JPMJFR2331，JST ムーンショット型研究開発事業 JPMJMS2011-35 (fundamental research)，および，文部科学省の補助事業「生成 AI モデルの透明性・信頼性の確保に向けた研究開発拠点形成」の助成を受けたものです.

参考文献

[1] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In **International Conference on Learning Representations (ICLR)**, 2023.

[2] MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, **Computer Vision – ECCV 2024 - 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXV**, Vol. 15133 of **Lecture Notes in Computer Science**, pp. 270–287. Springer, 2024.

[3] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In **Advances in Neural Information Processing Systems**, 2023.

[4] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are Super Mario: Absorbing abilities from homologous models as a free lunch. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, **Proceedings of the 41st International Conference on Machine Learning**, Vol. 235 of **Proceedings of Machine Learning Research**, pp. 57755–57775. PMLR, 2024.

[5] Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. MetaGPT: Merging large language models using model exclusive task arithmetic. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, **Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing**, pp. 1711–1724, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[6] Ondrej Bohdal, Mete Ozay, Jijoong Moon, Kyenghun Lee, Hyeonmok Ko, and Umberto Michieli. Efficient compositional multi-tasking for on-device large language models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, **Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing**, pp. 28129–28153, Suzhou, China, November 2025. Association for Computational Linguistics.

[7] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In **International Conference on Learning Representations (ICLR)**, 2022.

[8] Jinghan Zhang, Shiqi Chen, Junteng Liu, and Junxian He. Composing parameter-efficient modules with arithmetic operations. In Alice Oh, Thomas Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, **Advances in Neural Information Processing Systems**, Vol. 36, pp. 12589–12610. Curran Associates, Inc., 2023.

[9] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In **Advances in Neural Information Processing Systems**, Vol. 36, 2023. NeurIPS 2023 (Main Conference Track).

[10] Hongkang Li, Yihua Zhang, Shuai Zhang, Pin-Yu Chen, Sijia Liu, and Meng Wang. When is task vector provably effective for model editing? A generalization analysis of nonlinear transformers. In **International Conference on Learning Representations (ICLR)**, 2025. Oral.

[11] Luca Zhou, Daniele Solombrino, Donato Crisostomi, Maria Sofia Bucarelli, Giuseppe Alessio D'Inverno, Fabrizio Silvestri, and Emanuele Rodolà. On task vectors and gradients. In **NeurIPS 2025 Workshop: UniReps (Unifying Representations in Neural Models)**, 2025.

[12] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, **Proceedings of the 39th International Conference on Machine Learning**, Vol. 162 of **Proceedings of Machine Learning Research**, pp. 23965–23998. PMLR, 2022.

[13] Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. LoRA soups: Merging LoRAs for practical skill composition tasks. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, Steven Schockaert, Kareem Darwish, and Apoorv Agarwal, editors, **Proceedings of the 31st International Conference on Computational Linguistics: Industry Track**, pp. 644–655, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[14] Alex Gittens, Dimitris Achlioptas, and Michael W. Mahoney. Skip-gram − Zipf + uniform = vector additivity. In Regina Barzilay and Min-Yen Kan, editors, **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 69–76, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[15] Carl Allen and Timothy Hospedales. Analogies explained: Towards understanding word embeddings. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, **Proceedings of the 36th International Conference on Machine Learning**, Vol. 97 of **Proceedings of Machine Learning Research**, pp. 223–231. PMLR, 09–15 Jun 2019.

[16] Tim Van de Cruys. Two multivariate generalizations of pointwise mutual information. In Chris Biemann and Eugenie Giesbrecht, editors, **Proceedings of the Workshop on Distributional Semantics and Compositionality**, pp. 16–20, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[17] William J. McGill. Multivariate information transmission. **Psychometrika**, Vol. 19, No. 2, pp. 97–116, June 1954.

[18] Satosi Watanabe. Information theoretical analysis of multivariate correlation. **IBM Journal of Research and Development**, Vol. 4, No. 1, pp. 66–82, January 1960.

[19] Alexandra Chronopoulou, Jonas Pfeiffer, Joshua Maynez, Xinyi Wang, Sebastian Ruder, and Priyanka Agrawal. Language and task arithmetic with parameter-efficient layers for zero-shot summarization. In Jonne Sälevä and Abraham Owodunni, editors, **Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)**, pp. 114–126, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

A　参考情報

A.1　誤差項の分解の導出

加法構成性の誤差は式 (5)–(8) のように分解される．このことを導く．

タスク $i \in \{1, 2, *\}$ を考える．タスクを解く際にモデルに与える指示文を I_i，重み $\boldsymbol{\theta}$ のモデルの出力分布を p_θ とする．記法を簡潔にするために BOS トークンは省略すると，出力確率の対数は次のように変形される：

$$\log p_\theta(y|I_*, x) = \log p_\theta(y|I_1, x) + \log p_\theta(y|I_2, x) + \log \frac{p_\theta(y|I_*, x)}{p_\theta(y|I_1, x)p_\theta(y|I_2, x)}. \tag{10}$$

各タスクの損失がクロスエントロピー $L_i(\boldsymbol{\theta}) = -\mathop{\mathbb{E}}_{q_i}[\log p_\theta(y|I_i, x)]$ であるとき，上式の両辺をタスク $*$ のデータ分布 q_* で平均すると，

$$L_*(\theta) = -\mathbb{E}_{q_*}[\log p_\theta(y|I_1, x)] - \mathbb{E}_{q_*}[\log p_\theta(y|I_2, x)] - \mathop{\mathbb{E}}_{q_*}\left[\log \frac{p_\theta(y|I_*, x)}{p_\theta(y|I_1, x)p_\theta(y|I_2, x)}\right]. \tag{11}$$

右辺第 1 項は以下のように変形できる：

$$\begin{aligned}
-\mathbb{E}_{q_*}[\log p_\theta(y|I_1, x)] &= -\sum_{x,y} q_*(x, y)\ \log p_\theta(y|I_1, x) \\
&= -\alpha_1 \sum_{x,y} q_1(x, y)\ \log p_\theta(y|I_1, x) - \sum_{x,y} \big(q_*(x, y) - \alpha_1 q_1(x, y)\big)\ \log p_\theta(y|I_1, x) \\
&= \alpha_1 L_1(\boldsymbol{\theta}) - \sum_{x,y} \big(q_*(x, y) - \alpha_1 q_1(x, y)\big)\ \log p_\theta(y|I_1, x).
\end{aligned} \tag{12}$$

右辺第 2 項も同様である．

以上をまとめると次が得られる：

$$\begin{aligned}
\mathcal{C}(\boldsymbol{\theta}) = &-\sum_{x,y} \big(q_*(x, y) - \alpha_1 q_1(x, y)\big)\ \log p_\theta(y|I_1, x) - \sum_{x,y} \big(q_*(x, y) - \alpha_2 q_2(x, y)\big)\ \log p_\theta(y|I_2, x) \\
&- \mathop{\mathbb{E}}_{q_*}\left[\log \frac{p_\theta(y|I_*, x)}{p_\theta(y|I_1, x)p_\theta(y|I_2, x)}\right]
\end{aligned} \tag{13}$$

以上より，式 (5)–(8) が導出される．

A.2　構成的タスクにおける誤差項の分解

指示文 I_* がタスク 1 と 2 の指示文 I_1, I_2 の連結である場合，項 $\boldsymbol{\delta}_*$ は式 (9) のように分解される．以下に導出を示す．

ここでは，BOS トークンを省略せずに記載する．$p(BOS) = 1$ であることに注意して，以下が導ける：

$$\log \frac{p_\theta(y|BOS, I_*, x)}{p_\theta(y|BOS, I_1, x)p_\theta(y|BOS, I_2, x)} = -\log p_\theta(y|BOS, x) + SI(I_1; I_2; x|BOS) - SI(I_1; I_2; x, y|BOS), \tag{14}$$

$$SI(I_1; I_2; x|BOS) = \log \frac{p_\theta(x|BOS, I_1)p_\theta(x|BOS, I_2)}{p_\theta(x|BOS, I_1, I_2)} \frac{1}{p_\theta(BOS, x)}, \tag{15}$$

$$SI(I_1; I_2; x, y|BOS) = \log \frac{p_\theta(x, y|BOS, I_1)p_\theta(x, y|BOS, I_2)}{p_\theta(x, y|BOS, I_1, I_2)} \frac{1}{p_\theta(BOS, x, y)}. \tag{16}$$

式 (14) を微分し，分布 q_* のもとでの期待値をとることで，式 (9) が導かれる．