モデル編集を用いた Machine Unlearning における ハイパーパラメータの自動調節

中屋和樹 [1] 松田源立 [1]
[1] 成蹊大学大学院 理工学研究科
dm226206@cc.seikei.ac.jp matsuda@st.seikei.ac.jp

概要

深層学習の発展によりあらゆるタスクの性能が向上した一方で，特定のデータを忘れる技術の開発も求められるようになってきている。Machine Unlearning とは訓練された機械学習モデルから，一部のデータに関する影響を取り除く技術である。本研究では，タスクベクトルと呼ばれるモデル編集手法に着目し，Transforemr モデルの重みを一律ではなく個々の Attention ブロック，ないしはレイヤーごとに調節する手法を提案した。具体的には，翻訳タスクと対話システムタスクでの Unlearning を試み，Machine Unlearning における評価基準の観点から，テストデータおよび忘却データを対象とした分析を行った。

1 はじめに

近年，深層学習の発展により，機械学習と人工知能の分野は目覚ましい進化を遂げている。特に 2017 年に登場した Transformer[1] は，機械翻訳やその他の自然言語処理タスクだけでなく，画像処理や音声処理分野においても顕著な改善をもたらしている。しかし，この飛躍的な進歩と並行して，様々な潜在的問題も明らかになってきている。基本的に，機械学習モデルは，主にインターネットから収集した，大規模なデータセットを利用して学習を行う。この過程において，学習者の意図しないところでユーザーのプライバシーや著作権を侵害するリスクが生じる可能性がある。実際，2021 年には，韓国製のチャットシステムが顧客の住所や銀行口座番号を意図せずに生成したことで，個人情報保護違反で起訴された事例が存在する。[2] また，Stable Diffusion のような画像生成 AI と著作権との問題は非常に根深い。これは，AI が生成する画像が既存の著作物から派生しているか，それとも独自の創造物であるかの判断が困難なためである。AI によって生成された画像が，既存の著作物の表現上の本質的な特徴を捉えている場合，それは著作権の侵害と見なされる可能性がある [3]。これらの情勢を踏まえ，プライバシーや著作権保護の観点から，機械学習モデルが以前に学習した情報を忘れる，すなわち「データ忘却技術」の開発がより重要視されてきている。Machine Unlearning とは訓練された機械学習モデルから，一部のデータに関する影響を取り除く技術である。[4] 学習モデルが有害なテキストを出力することを抑える Detoxify[5][6] や，モデルに内在するジェンダーバイアスなどを除去する DeBias[7][8] は，Machine Unlearning におけるタスクの一部である。

[9] では，事前学習済みモデル同士の重みを算術演算することで，その振る舞いを変更する「タスクベクトル」というアプローチを提案している。タスクベクトルは，あるタスクでファインチューニングされたモデルの重みから事前学習済みモデルのパラメータの重みを引くことで得られ，加算や減算などの算術演算を通じてターゲットタスクに関するモデルの編集が可能とされている。本研究ではタスクベクトルにおける，negation(否定) に焦点を当てる。否定を表現するタスクベクトルは通常のタスクベクトルにマイナス符号をつけることで得られ，これを編集対象のモデルに指定した割合だけ加算することで，Unlearning 操作を実現できる。原著論文では加算割合 λ はハイパーパラメータとなっており，また，全ての重みに対して一律に適用している。Transformer は Attention ブロックを複数積み重ねた構造をしており，個々のブロック，ないしはレイヤーが保有している忘却対象データの情報は異なる可能性がある。本稿では，個々の Attention ブロックとレイヤーについてハイパーパラメータ λ を調節する手法を提案し，汎化性能の維持と Unlearning 性能の 2 つの観点から比較を行った。

2 関連研究

本節以降，全ての学習データを D_{all}，忘却対象のデータを D_{forget}，情報を維持するデータを D_{remain}，テストデータを D_{test} と表記する。通常の学習モデルを M，Unlearning 適用モデルを UM，また，データ D を用いてモデルを訓練したことを $M(D)$ で表す。ここでそれぞれの D は，以下の条件を満たすとする。

- $D_{all} = D_{forget} \cup D_{remain}$
- $D_{forget} \cap D_{remain} = \emptyset$

$M(D_{remain})$ は D_{all} から D_{forget} を除外して学習したモデルであり，本論では Retrain(再学習) モデルという呼称で表現し，UM モデルとの振る舞いの差を評価する際に用いる。

2.1 Machine Unlearning の達成条件

本節では，Machine Unlearning が達成されたと判断する基準について検討する。既存の研究 [4] によれば，Machine Unlearning の目標は，D_{forget} の影響を学習モデルから除去することにある。重要な点は，これが D_{forget} に対するモデルの精度を際限無く低下させることを意味するわけではないということである。例として，D_{remain} のみを用いてモデルを学習する場合を考える。このモデル ($M(D_{remain})$) は，D_{forget} を未観測であるため，$M(D_{all})$ とは異なる挙動を示し，擬似的に Unlearning を行っていると考えられる。この観点から，本稿では，Machine Unlearning が達成されるための十分条件を次のように定義する。

1. D_{forget} に対する $M(D_{remain})$ と UM の振る舞いが近しい
2. D_{test} に対する UM の精度が維持されている[1)]

2.2 Unlearning 手法の評価基準

Unlearning 手法の評価基準に関しては多岐にわたる議論が存在する。再学習に要する時間は，Unlearning 手法の効率性を評価する上で有効な指標と考えられる。[10] しかしながら，Unlearning に必要な補助モデルの学習時間や，ハイパーパラメータのチューニングに伴う時間的なばらつきを考慮すると，再学習時間の一律な測定は困難であり，本研究ではこの指標を採用していない。Machine Unlearning の分野においては，目的のタスクに応じた性能スコアが広く用いられている。[10][11] 本研究も，このアプローチを踏襲する。さらに，言語モデルが単語列に対する確率分布を生成する特性を考慮し，単語分布の類似性を Unlearning 手法の評価基準として採用する。確率分布間の差異を測定する手段として，対称性を持つ Jensen-Shannon 情報量を採用した。

2.3 タスクベクトルによるモデル編集

タスクベクトルとは，[9] で提案された，事前学習済みモデルの挙動や下流タスクにおけるモデルの性能を制御するための手法である。タスクベクトルは，特定のタスクでファインチューニングさせたモデルから，事前学習モデルの重みを引くことで構成される。本研究ではこの考え方を応用し，忘却対象のデータに特化してファインチューニングさせたモデルから全てのデータでファインチューニングさせたモデルの重みを引くことで，タスクベクトルとして用いる。また，Unlearning の際には，タスクベクトルにマイナス符号をつけて否定表現に変換し，全てのデータで学習させたモデルに一定の割合で足し合わせる。以上の流れを数式で表すと以下の通りとなる。まず，タスクベクトルを以下のように構成する。

$$\tau_{task} = \theta_{Dforget} - \theta_{Dall} \tag{1}$$

$\theta_{D_{forget}}$，$\theta_{D_{all}}$ はそれぞれ，D_{all} でのファインチューニング+D_{forget} で追加のファインチューニングを行ったモデルの重み，D_{all} でファインチューニングを行ったモデルの重みを表している。続いて，以下の式に従って Unlearning を行う。

$$\theta_{new} = \theta_{Dall} - \lambda\, \tau_{task} \tag{2}$$

ここで，λ は範囲が 0〜1 の scaling パラメータであり，否定表現のタスクベクトルを足し合わせる割合を示す。

3 提案手法

本節では提案手法の詳細について述べる。[9] で提案されている Unlearning は，タスクベクトルにマイナス符号を付けて否定表現に変換し，変換されたベクトルを編集対象のモデルに λ の割合で加算する，という過程を経て実行される。ただし，λ はハイパーパラメータであり，手動での調整が必要となる。本研究では，この λ を自動で調節することを目

1) 2 について，本来は，D_{remain} についての精度が維持されていることが条件であるが，計算時間の都合上，今回は D_{test} で代用した。

表 1: Layer Dropping した時のスコア (IWSLT は BLEU，Persona は Perplexity)

	Encoder						Decoder						
Layer	1	2	3	4	5	6	1	2	3	4	5	6	Not drop
IWSLT	14.38	43.21	45.89	46.99	46.61	18.13	12.36	23.34	45.25	46.62	43.02	18.83	50.52
Persona	7.01	6.64	6.72	6.64	6.66	331.89	5734.46	46.41	19.41	7.98	8.70	321.41	6.35
Persona(past)	7.36	5.89	5.86	5.88	6.08	1201.53	$4.0e^7$	35.79	14.3	6.80	7.60	335.34	5.39

指す。

3.1 手法 1: ブロックごとの割り当て

手法 1 では，Transfomrer の Attention ブロックを複数重ねた構造に着目し，個々のブロックが忘却対象データに対して個別の情報を持っていると仮定を置き，それに応じたハイパーパラメータの調節を行う。個々の Attention ブロックについて λ を調節するために，Layer Dropping を用いてスコアの変化量を調査する。Layer Dropping とは，複数の層が重なった Transofomer から Attention ブロックを削除して，最終的なスコアを求めることである。例えば，i 番目のブロックを削除した状態を出力を得た時に目的とするスコアが下がった場合，そのブロックには D_{forget} に関する情報が多く含まれており，減算する割合を増やすべきであると考えられる。[12] や [13] では，BERT や Vanilla Transformer を対象とした Layer Dropping による性能調査が行われている。i 番目のブロックの scaling パラメータを λ_i とすると，λ_i は以下の式で表される。

$$\lambda_i = 1 - \frac{score(drop(M(D_{all}), i), D_{forget})}{score(M(D_{all}), D_{forget})} \quad (3)$$

score(M，D) は，モデル M にデータ D を入力してスコアを得る操作を表しており，score は目的のタスクに応じた評価指標を採用する。翻訳タスクについては BLEU，対話システムタスクについては Perplexity を使用した。drop(M，i) は Transformer の i 番目のブロックを削除することを表現する関数である。

3.2 手法 2: レイヤーごとの割り当て

手法 1 では個々のブロックごとに λ を定めたが，手法 2 ではさらに個々のレイヤーに焦点をあて，細かい粒度での λ の調節を試みる。既存の研究 [7] において，勾配の類似度を用いた Debias 手法が提案されており，本研究でもこの考え方を応用する。$M(D_{all})$ および $M(D_{forget})$ について，D_{forget} の勾配を各々のレイヤーごとに比較した際，類似度が高いレイヤーが存在することを仮定する。そして，その類似度に応じて λ を設定する。具体的には，$M(D_{all})$ と $M(D_{forget})$ に D_{forget} を入力し，得られた勾配のレイヤーごとの類似度を求めて λ とする。本研究では cos 類似度を採用した。手法 1 の考え方と同様に，類似度が高いレイヤーについては，D_{forget} に関する情報が多く含まれており，減算する割合を増やすべきであるとする。i 番目のレイヤーの λ_i は以下の式の通り。

$$\lambda_i = \frac{1}{|D_{forget}|} \sum_{d \in D_{forget}} \frac{\nabla_i^{(M_1,d)} \cdot \nabla_i^{(M_2,d)}}{\| \nabla_i^{(M_1,d)} \| \| \nabla_i^{(M_2,d)} \|} \quad (4)$$

ここで M_1 は $M(D_{all})$，M_2 は $M(D_{forget})$ である。また，$\nabla_i^{(M,d)}$ はモデル M にデータ d を入力した際の勾配ベクトルを表す。実装においては，各勾配はバッチごとに求めるため，平均は$|D_{forget}|$の代わりに D_{forget} の全バッチ数での除算によって求める。

4 実験設定

実験データ，およびアーキテクチャの詳細については Appendix A にまとめている。

5 結果と考察

5.1 Dropping によるスコア変化

表 1 に，Layer Dropping を実行した際の D_{forget} のスコアを示す。IWSLT データセットに関しては，Layer Dropping を適用した結果, encoder と decoder の両方で BLEU スコアに顕著な変化が観察された。具体的には，encoder と decoder の両方で 1 層目と 6 層目を削除した際の BLEU スコアの低下が目立った。加えて，decoder では 2 層目を削除した際の BLEU スコアも大幅に低下した。以上より，encoder の 1 層目と 6 層目，decoder の 1 層目，2 層目，6 層目には，D_{froget} に関する情報がより多く含まれている可能性が示唆される。

Personachat データセットに関しては，encoder と比較して decoder での perplexity の変化がより大きいことが明らかになった。過去の対話履歴の有無に関わらず，encoder の 1 層目から 5 層目をそれぞれ削

表 2: 各データセットにおける Unlearning の結果 (PPL と JSD はスコアが低いほど良い。)

Models	Test Set						Forget Set					
	IWSLT		Persona		Persona(past)		IWSLT		Persona		Persona(past)	
	BL4	JSD	PPL	JSD	PPL	JSD	BL4	JSD	PPL	JSD	PPL	JSD
Original	34.28	-	19.84	-	18.54	-	38.17	-	12.33	-	10.82	-
Retrain	34.49	-	19.73	-	18.08	-	34.99	-	18.26	-	16.60	-
Model Edit	**33.86**	**0.10**	**19.82**	**0.11**	**18.32**	**0.12**	**35.01**	0.10	**18.60**	**0.11**	**16.64**	0.12
Model Edit + Layer Drop	**33.86**	0.17	20.05	0.13	18.60	0.13	34.35	0.20	19.77	0.13	18.05	0.14
Model Edit + Grad Sim	33.21	0.13	20.37	0.32	18.72	0.15	35.80	**0.09**	15.56	0.15	13.83	**0.11**

除した際の Perplexity の変化は軽微であり，6 層目を削除した場合に大幅な変化が観察された。一方，decoder では 1 層目と 6 層目による Perplexity の大幅な変化が観察された。特に，過去の対話履歴ありで 1 層目を削除した場合，4.0e^7 ポイントという，より急激な上昇となっている。2 層目から 5 層目においては，4 層目と 5 層目が比較的軽微であり，2 層目と 3 層目が中程度の上昇を示した。これらの結果から，encoder の 6 層目，decoder の 1 層目，2 層目，3 層目，6 層目には D_{forget} に関連する情報がより多く含まれている可能性が示唆される。Appendix B の図 1 と図 2 に，表 1 の数値から求めた各レイヤー別の λ_i のグラフを載せている。

5.2 Unlearning の定量的評価

表 2 について，Original と Retrain はそれぞれ，全てのデータで学習したモデル ($M(D_{all})$)，再学習モデル ($M(D_{remain})$) を表している。BL4 は 4-gram の BLEU スコア, PPL は Perplexity, JSD は Jensen-shanon Divergence である。JSD は，再学習モデルと UM との間で計算した。Model Edit を実行する際の scaling パラメータ λ は，Retrain モデルにおける forget set のスコアに近づくようにチューニングを行った。具体的なパラメータの値は次の通り。IWSLT:0.36, Persona: 0.57, Persona(past): 0.55。

まず，Test Set の結果に関して概観する。IWSLT の BL4 については，Layer Drop が手動チューニング (Model Edit) と同等のスコアを示したが，その他のデータセット・指標については手動チューニングが最も高い性能となっている。また，Layer Drop と Grad sim での比較では全てのデータセット・指標で Layer Drop が高い性能を示した。これは，手動チューニングで単一の λ を設定する方法が，最も汎化性能を維持することを示唆している。

続いて，Forget Set の結果について概観する。IWSLT データセットでは，手動チューニングが BLEU スコアにおいて Layer Drop，Grad sim を上回り，JSD では Grad Sim が手動チューニングより 0.01 ポイント低い結果となった。一方，Personachat データセットでは，過去の対話履歴を考慮した場合に限り，Grad sim の JSD が手動チューニングよりも 0.01 ポイント低い結果を示した。その他の結果に関しては，全て手動チューニングが優れていた。Perplexity は対話履歴ありとなしのそれぞれにおいて，Layer Drop は過剰に Unlearning しており，Grad sim は Unlearning が不足していることが分かる。

これらの結果を総合すると，今回の実験においては，λ を個々のブロックないしはレイヤーごとに設定するよりも，手動チューニングによる単一の設定がより優れたスコアを示し，提案した手法では手動でのチューニングには及ばない結果となった。また，性能スコアの高さと単語分布間の距離の近さが必ずしも比例関係にないことも明らかになった。

6 おわりに

本研究では，タスクベクトルというモデル編集手法に焦点を当て，ハイパーパラメータを Layer Dropping，および勾配の類似度を用いて調整する手法を提案した。実験結果から，Transformer の各ブロックが D_{forget} について異なる情報を保持している可能性が示唆された。また，提案手法では手動でのチューニングには及ばない結果となったが，精度の高さと単語分布間の距離の近さが必ずしも比例関係にないという興味深い発見が得られた。今後の研究課題としては，異なるアプローチによる λ の調節方法の探索や新しい Unlearning 手法の開発が挙げられる。また，そもそも再学習モデルとはどのような存在なのか？それらの本質的な性質についての分析も重要な研究課題となるだろう。

謝辞

本研究は JSPS 科研費 JP21K12036 の助成を受けたものである。

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. **CoRR**, abs/1706.03762, 2017.

[2] Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. 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 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics.

[3] Midjourney、Stable Diffusion、mimic などの画像自動生成 AI と著作権（その 2），（2023-12 閲覧）. https://storialaw.jp/blog/8883.

[4] Thanveer Shaik, Xiaohui Tao, Haoran Xie, Lin Li, Xiaofeng Zhu, and Qing Li. Exploring the landscape of machine unlearning: A comprehensive survey and taxonomy, 2023.

[5] Zecheng Tang, Keyan Zhou, Pinzheng Wang, Yuyang Ding, Juntao Li, and Minzhang. Detoxify language model step-by-step, 2023.

[6] Shrimai Prabhumoye, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Adding instructions during pretraining: Effective way of controlling toxicity in language models. In Andreas Vlachos and Isabelle Augenstein, editors, **Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics**, pages 2636–2651, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[7] Charles Yu, Sullam Jeoung, Anish Kasi, Pengfei Yu, and Heng Ji. Unlearning bias in language models by partitioning gradients. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, **Findings of the Association for Computational Linguistics: ACL 2023**, pages 6032–6048, Toronto, Canada, July 2023. Association for Computational Linguistics.

[8] Masahiro Kaneko and Danushka Bollegala. Debiasing pre-trained contextualised embeddings. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, **Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume**, pages 1256–1266, Online, April 2021. Association for Computational Linguistics.

[9] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In **The Eleventh International Conference on Learning Representations**, 2023.

[10] Lingzhi Wang, Tong Chen, Wei Yuan, Xingshan Zeng, Kam-Fai Wong, and Hongzhi Yin. KGA: A general machine unlearning framework based on knowledge gap alignment. In **Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pages 13264–13276, Toronto, Canada, July 2023. Association for Computational Linguistics.

[11] Vikram S. Chundawat, Ayush K. Tarun, Murari Mandal, and Mohan S. Kankanhalli. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In Brian Williams 0001, Yiling Chen 0001, and Jennifer Neville, editors, **Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023**, pages 7210–7217. AAAI Press, 2023.

[12] Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. On the effect of dropping layers of pre-trained transformer models, 2022.

[13] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In **Advances in Neural Information Processing Systems**, volume 32. Curran Associates, Inc., 2019.

[14] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In Iryna Gurevych and Yusuke Miyao, editors, **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pages 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[15] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

A　実験設定

A.1　実験データ

翻訳タスクについては IWSLT2014 の英語-ドイツ語の対訳コーパスを採用した。D_{forget} については，train データのうち，0.1%をランダムに選択した。

対話システムについては PersonaChat コーパス [14] を使用した。これらのデータセットから，「過去の履歴を参照したデータセット」と「過去の履歴を参照しないデータセット」を別々に構築した。D_{forget} については，convid からランダムに 200 個の id を選択した。Personachat データセットは personality，candidates，history，convid，utteranceidx のカラムから構成されており，それぞれ以下の情報が格納されている。

Personality: 話者の性格を記述した文章。
Candidates: 話者の性格とこれまでの会話の履歴を考慮に入れた，話者が発した真の発言と誤答の発言を含む発言リスト。最後の発話が正解となる発話である。
History: 話者同士の対話の履歴。
convid: 対話のテーマごとに付与された id。
utteranveidx: 個々の対話に紐づけられた id。

過去の履歴を参照する場合は，Hisory のテキストを </s>トークンで結合し input とし，Candidates の最後のテキストを output とした。過去の履歴を参照しない場合は，History の最後のテキストを input とし，Candidates の最後のテキストを outoput とした。

表 3: データセットの統計量

	IWSLT	Persona	Persona (past)
D_{all}	160,239	131,438	131,438
D_{test}	6,750	7,801	7,801
D_{forget}	1,603	1,395	1,395
D_{remain}	158,636	130,043	130,043
Avg of input	40.91	11.89	94.51
Avg of output	21.53	12.36	12.36

A.2　アーキテクチャ

本研究では，学習モデルとして BART[15] を選択した。BART は，Meta によって開発された encoder-decoder 構造を持つ Transformer ベースのアーキテクチャである。このモデルの encoder 部分は，BERT における学習手法と同様，Masked Language Modeling を用いて一部の単語をマスクして学習を行う。一方で，decoder 部分は，次に続く単語の予測を目的とした学習を実施する。文章生成を行う際には，t-1 番目までのテキストを入力として用い，自己回帰形式により t 番目の単語を予測する。文章の最大トークン長は IWSLT，Perosna が 128，Persona(past) が 256 とした。オプティマイザは AdamW を使用し，学習率は 2e-5 に設定した。学習エポック数は 3 とした。

B　各 Attention ブロックの λ

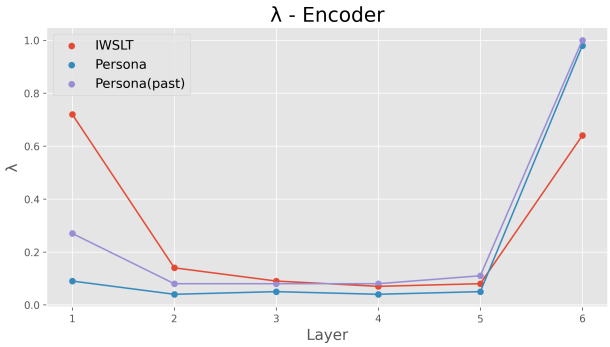

図 1: Encoder Attention ブロックの λ_i

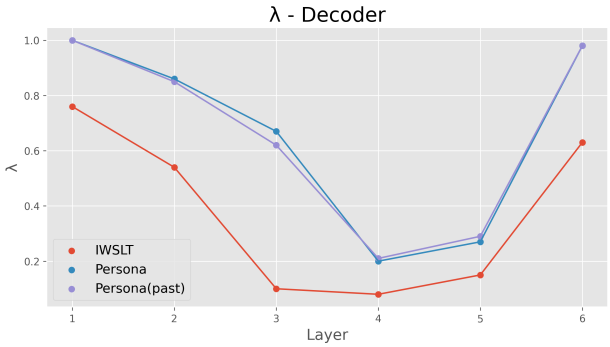

図 2: Decoder Attention ブロックの λ_i