知識志向 Mixture of LoRA Experts の構築

伊藤俊太朗　河原大輔
早稲田大学理工学術院
schnellheiter@fuji.waseda.jp　dkw@waseda.jp

概要

大規模言語モデル（LLM）の進化は驚異的であるが、知識の管理には依然として課題が存在する。全パラメタの再学習には膨大なリソースが必要なため、LoRA という手法が採用されることが多い。さらに、複数モデルの出力を組み合わせること（Mixture of Experts; MoE）による性能向上が注目されている。本研究では、LoRA と MoE を組み合わせて、モデルのパラメタ増加を最小限に抑えつつ性能を向上させる方法を提案する。提案手法は、知識を専門家（expert）に依存させることで、知識をより効果的かつ局所的に管理する。CommonsenseQA データセットを利用した評価によって提案手法の有効性を確認した。

1　はじめに

大規模言語モデル (LLM) の進化は驚異的であり、対話形式で指示を入力すれば内容に応じた様々な応答を柔軟に生成することが可能となった。初期の Decoder-only モデルである GPT [1] のパラメタ数は 0.1B 程度であったが、モデルのパラメタ数および学習データ量を増量することでモデルが様々なタスクを汎用的にこなせることが明らかになった [2]。その後、モデルのパラメタ数や学習データ量をさらに増やす傾向が進んだ。GPT-3 [3] では 175B ものパラメタを持つようになり、少数の回答例を入力に含めることで、これまで fine-tuning が必要であったタスクにおいてもパラメタを更新せずに回答可能になった。

GPT-2 [2] の学習には 22GB 程度[1)]の GPU メモリが必要となる。学習に使うデータや内部状態の容量も考慮すればさらに大規模なモデルを扱う場合一般的に用いられている GPU では学習が難しい。そこでモデルの追加学習時に更新するパラメタを一部に絞ることで学習時に必要となる GPU メモリを大幅に減らす手法である LoRA [4] が提案された。

また、LLM の登場により単一のモデルで幅広いタスクへの対応が可能になり、モデルが想定する入力空間が膨大になる傾向が進んだ。そこで、expert と呼ばれる様々なモデルの出力を組み合わせることで、各々の expert を入力空間の異なる部分に特化させるアプローチ Mixture of Experts (MoE) [5] が注目されるようになった。

これらの手法は、汎用モデルに追加の知識を付与する場合にも有効である。しかし、LoRA 単体では追加知識が 1 組の追加パラメタに全て付与されるため、新たな知識や以前の学習で記録した知識を保持することが難しい。また、MoE を利用する場合は expert ごとに独立したモデルの出力を結合するため、expert で利用するモデルの大規模化がハードウェアの制約で難しくなる。本研究では、LoRA と MoE を組み合わせることで、追加知識の管理の煩雑化やパラメタの上昇を抑えつつ新たな知識を付与する手法を提案する。CommonsenseQA データセットを利用した評価によって提案手法の有効性を確認した。

2　関連研究

2.1　LoRA

LoRA (Low-Rank Adaptation) [4] は、特に大規模な言語モデルにおけるパラメタの全体的な微調整の限界に対処するために提案された。従来の全体的な微調整では、各ダウンストリームタスクごとに異なるパラメタを学習する必要があり、効率が悪くリソースを大量に消費する問題があった。LoRA により、タスク固有のパラメタの差分を低ランク行列で表現することで、特に大規模なモデルにおいて効率的な微調整が可能となった。

LoRA の主なアイデアは、ニューラルネットワークの密な層で使用される低ランクパラメタ化更新行列の利用にある。通常のニューラルネットワーク

1）　32bit での学習を想定

では、これらの密な層は一般にフルランクの重み行列を使用して行列乗算を行う。LoRA は、タスク適応中のこれらの重みの更新が低い「固有ランク」を持っていると仮定し、この低ランクアプローチにより、フルランクの更新を必要とせずに、全体的な微調整の表現力を維持することが可能となる。その結果、LoRA は Transformer [6] のようなニューラルネットワーク内の任意の重み行列のサブセットに適用でき、訓練可能なパラメタの数を減少させることができる。

2.2 Mixture of Experts

Mixture of Experts (MoE) [5] は、機械学習におけるアンサンブル手法の一つであり、特定の問題に対して最適な解を提供するために、複数の専門家（小規模なモデル, expert）が共同で作業するアプローチである。MoE では、各専門家は問題の特定の側面に特化し、ゲート関数が入力に基づいて最も適切な専門家を動的に選択する。これにより、アンサンブルの中で最も知識のある専門家の意見が重視され、全体としての性能が向上する。この手法の導入により、より柔軟かつ適応性の高いモデル構築が可能となり、特に多様な特徴やパターンを持つデータに対して効果を発揮する。

MoE の主要な構成要素として expert とゲート関数があり、これらをどのように設計するかで様々な派生が存在する。自然言語処理における応用例としては、翻訳タスクにおいて事前に大量の expert をゲート関数によって疎に結合し計算コストを抑えつつ擬似的なパラメタ数を大幅に増強する手法 [7] がある。また、expert を LoRA で代替してゲート関数により疎に出力を結合することでパラメタ数を抑えつつアンサンブル学習の効果による性能向上を狙う手法 [8] がある。

2.3 CommonsenseQA

質問に回答する際に人間は文脈だけではなく世界知識を活用する。このような事前知識を伴う質問応答能力を測定するために CommonsenseQA [9] が構築されている。これは、ConceptNet [10] からした抽出したエンティティとその類似エンティティを判別するための多肢選択式質問をクラウドワーカに作成してもらうことによって構築された。これらの質問に回答するにはコンテキスト外の事前知識が必要となる。例えば図 2 において正しい回答 “complete job” を選択するには、与えられた文脈だけでなく “work” という単語に関する世界知識が必要となる。

本研究では、expert への知識付与の効果検証としてこのデータセットを利用して評価を行う。

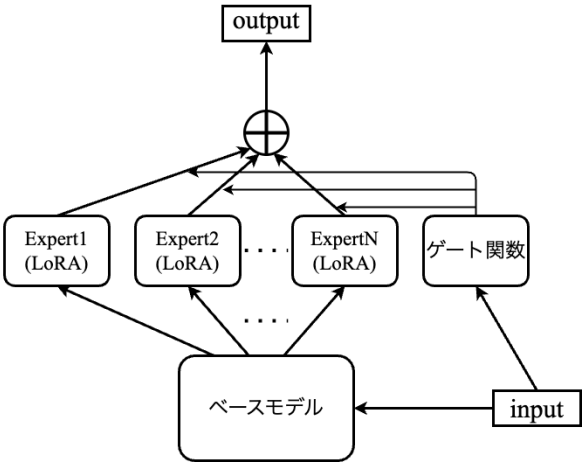

図 1 提案手法の概要図

3 提案手法

MoE は機械学習モデルの性能を上げる手法として一般的に知られている。しかし、モデルが大規模化する近年の傾向を考えると、主に計算コストの問題で複数の大規模モデルを用意して運用するのは難しい。本研究では、特定のデータセットやドメインに特化した expert を LoRA で学習し、ゲート関数を用いて出力を重み付けして結合する手法を提案する。これにより、必要な知識を局所化しつつ、モデル全体のパラメタの増加が抑制される。手法の概要図を図 1 に示す。モデルの構成要素のうち、ベースモデルと expert は独立して学習を行い、ゲート関数はモデル全体の出力調整のための追加学習時に学習を行う。

3.1 ベースモデル

ベースモデルは全ての expert (LoRA) の基盤となるモデルである。通常の MoE ではそれぞれの expert が独立したモデルを利用するのに対し、提案手法においては全ての expert においてこのベースモデルが共通のパラメタとなる。そのため、推論及び学習時に利用される GPU メモリは expert として独立モデルを利用する場合と比べて大幅に削減され、近年の大規模なモデルに対しても容易に適用できる。なお、追加学習時においてベースモデルのパラメタは全て凍結させる。

3.2 Expert

各 expert の LoRA パラメタは特定のタスクやデータセットに特化するように予め学習する。学習済みの LoRA パラメタの挿入によりベースモデルの状態からパラメタ調整が入ったモデルを expert として扱う。LoRA パラメタは追加学習時にも更新する。

3.3 ゲート関数

ゲート関数は、各 expert の出力をどの程度の比重で結合するかを決定する重みを動的に作成するために利用する。入力として想定される自然言語データ x は離散表現であり、直接重み行列への変換ができない。そこで BERT [11] 等の小規模な encoder モデルで連続表現 $H(x)$ に変換した後に、次元調整用の行列 W を乗算した上で Softmax 関数で調整したものを重み $G(x)$ とする。その重みを利用して各 expert の出力 $E_i(x)$ を結合したものを提案手法の MoE モデルの出力 $O(x)$ とする。

$$G(x) = \text{Softmax}(\text{H}(x) \cdot W) \tag{1}$$

$$O(x) = \sum_{i}^{N} G_i(x) \cdot E_i(x) \tag{2}$$

4 実験

提案手法の有効性を検証するために、複数の比較モデルを作成し CommonsenseQA データセットで性能の評価を行う。

4.1 データセット

データセットとして、expert に知識を付与するためのもの、および性能評価用のものの複数個を用意する。expert を事前に学習するための知識源としては、Wikipedia の情報を基に見出し語とその自然言語での説明をカテゴリごとにまとめたデータセットである DBPedia[2] [12] と、評価データである CommonsenseQA の構築に利用された ConceptNet[3] を利用する。DBPedia は見出し語の説明文を連結、ConceptNet はエンティティ同士の関係を説明する自然言語の文が存在する事例のみを抽出したもの連結し、それぞれトークナイズした上で各事例が 512 トークンになるようにしたものを expert 学習用のデータセットとする。

2) https://huggingface.co/datasets/dbpedia_14
3) https://huggingface.co/datasets/conceptnet5

```
Please answer the following questions from the options below.
Question: What do people aim to do at work?
Option 1: complete job
Option 2: learn from each other
Option 3: kill animals
Option 4: wear hats
Option 5: talk to each other
Answer:
```

図 2 プロンプトの例

性能評価は CommonsenseQA[4] を利用する。decoder モデルを利用するため、あらかじめ問題を図 2 のような形式のプロンプトに変換する。解答の生成は選択肢ではなく、選択肢の内容を生成する形式とする。なお、モデルが正解の選択肢と完全に一致する出力を行った場合のみ正答として扱い、正答率の評価を行う。例えば図 2 の問題の場合、“complete job” をモデルが生成した場合に正答として扱う。

4.2 モデルの学習設定

gpt2-xl[5] をベースモデルとして複数の LoRA モデルを作成し、提案手法の有効性を検証する。MoE モデルの expert の数は $N = 3$[6] とする。いずれのモデルにおいても、解答の形式を学習させるために CommonsenseQA の学習データを 3 epoch 程度利用して LoRA およびゲート関数のパラメタ更新を行う。そして、検証データを利用して正答率の評価を行う。比較モデルは、以下のとおり、ゲート関数を除いたネットワークのパラメタの数が一致するように LoRA の低ランク行列の階数を設定する。

- MoE (ConceptNet, DBPedia): ベースモデル、LoRA (r=8, ConceptNet)、LoRA (r=8, DBPedia) を expert とする MoE モデル
- MoE (random init): ベースモデルに加え、expert の LoRA (r=8) をランダムに初期化したものを 2 つ用意[7] した MoE モデル
- LoRA (r=16): ベースモデルに LoRA (r=16) を挿入し微調整したモデル
- LoRA (r=16, ConceptNet+DBPedia): ベースモデルに ConceptNet 及び DBPedia で追加学習した LoRA (r=16) を挿入し微調整したモデル

4) https://huggingface.co/datasets/commonsense_qa
5) https://huggingface.co/gpt2-xl
6) ハードウェアリソースの都合上の制限
7) 行列 A をランダムに初期化、行列 B を 0 で初期化

4.3 結果

CommonsenseQA の検証データセットにおける各モデルの正答率を表 1 に示す。また、学習過程における 100 ステップごとの正答率を図 3 に示す。

MoE モデル同士の比較では、提案手法である、事前に知識源を利用して学習する場合 (MoE (ConceptNet, DBPedia)) と、LoRA のパラメタをランダム初期化する場合 (MoE (random init)) を比較した際、提案手法が 6% 程度高い正答率となっている。また、図 3 から MoE (random init) の条件で学習した場合、過学習が発生していることが分かる。

ランク $r = 16$ の LoRA を利用して ConceptNet および DBPedia で学習したモデルを利用した場合 (LoRA (r=16, ConceptNet+DBPedia)) との比較では、提案手法は 3% 程度高い正答率となっている。なお、$r = 16$ の LoRA パラメタを利用して ConceptNet および DBPedia で事前知識の学習をしない場合は、正答率 20.6% となり最も低い正答率となった。

4.4 ゲート関数の事前学習の検証

ゲート関数はモデルへの入力テキストを基にして、どの expert の出力をどの程度利用するかを動的に決定する。提案手法においてはゲート関数、各 expert の LoRA を同時に学習しており、各 expert の LoRA は評価タスクの回答に役立つ知識で事前に学習されている。一方、ゲート関数に関しては事前学習されておらず、ゲート関数で入力テキストのエンコードに利用されるモデルが大規模テキストで事前学習されているのみである。

そこで、expert の perplexity を元にラベル作成を行い、ゲート関数の事前学習を行って検証する。ラベルは、i 番目の expert の perplexity を PPL_i として、以下のルールで作成する。各 expert の負数を softmax 正規化することで、最も perplexity の小さなモデルの出力を大きな割合で採用するように学習が進む。

$$\mathrm{label} = \mathrm{Softmax}(-\mathbf{PPL}(x)) \tag{3}$$

$$\mathbf{PPL}(x) = (\mathrm{PPL}_0(x), \mathrm{PPL}_1(x), \ldots) \tag{4}$$

全ての訓練事例を利用してラベル付を行いゲート関数を学習させた MoE (pretrained encoder full)、および訓練事例の 5%を利用した MoE (pretrained encoder partial) の学習過程を図 3 に示す。

最初の 100 ステップまでは提案手法よりも正答率が良いが、200 ステップ以降は全ての時点において提案手法の方が良い。この結果から、perplexity によるゲート関数の事前学習はモデルの性能に良い影響を与えないと言える。

表 1 CommonsenseQA における各モデルの正答率 (EM は完全一致による正答率を表す)

モデル	EM (%)
MoE (ConceptNet,DBPedia)	**28.1**
MoE (random init)	21.9
MoE (pretrained encoder full)	25.7
MoE (pretrained encoder partial)	25.6
LoRA (r=16)	20.6
LoRA (r=16, ConceptNet+DBPedia)	24.4

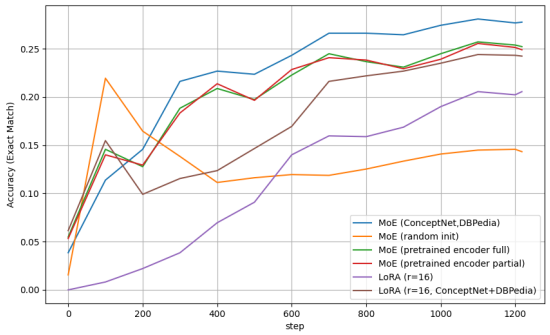

図 3 100 ステップごとの正答率

5 おわりに

本論文では LoRA と MoE を統合してモデルに新たな知識を付与する手法を提案した。知識志向で事前に調整した LoRA モデルの出力を結合することで知識の保存箇所を局在化しパラメタの増加を抑えつつ知識志向タスクの精度の向上が期待できる。貢献点をまとめると以下のとおりである。

- MoE アーキテクチャの expert として LoRA を利用することでパラメタの増加を最小限に抑えて MoE の構築が可能になる。
- ベースモデルのパラメタを凍結するので、ベースとなるモデルに含まれる知識は保存される。
- 追加知識が LoRA のパラメタに局在するので新たな知識を付与しやすくなる。

本研究の検証の範囲ではハードウェアの都合上、ベースモデルとして gpt2-xl の利用にとどまっている。今後は、さらに大規模なモデルをベースとして利用した場合の挙動の確認、また、知識付与用および評価用のデータセットをより関連性の高いものにし、より詳細な定性分析を行っていきたい。

謝辞

本研究は JSPS 科研費 JP21H04901 の助成を受けて実施した。

参考文献

[1] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

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

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[4] 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, 2021.

[5] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive Mixtures of Local Experts. **Neural Computation**, Vol. 3, No. 1, pp. 79–87, 03 1991.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[7] Noam M. Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. **ArXiv**, Vol. abs/1701.06538, , 2017.

[8] Yun Zhu, Nevan Wichers, Chu-Cheng Lin, Xinyi Wang, Tianlong Chen, Lei Shu, Han Lu, Canoee Liu, Liangchen Luo, Jindong Chen, and Lei Meng. Sira: Sparse mixture of low rank adaptation, 2023.

[9] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, **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. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[10] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. **Proceedings of the AAAI Conference on Artificial Intelligence**, Vol. 31, No. 1, Feb. 2017.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, **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.

[12] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, **The Semantic Web**, pp. 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

A　トークン生成時の設定

Hugging Face Transformers の generate メソッドを利用する。トークン生成にはビームサーチを利用し 3 トークン先までの生成結果を考慮した上で尤度の最も高いトークンを順に生成する。repetition penalty は設定せず、その他のハイパーパラメタは generate メソッドのデフォルトに従う。

B　学習時のハイパーパラメタの設定

学習には Transformers[8] の Trainer クラスを利用しここに記載していないハイパーパラメタについてはデフォルト値を利用している。また全ての実験においてデバイス数は 3 である。

表 2　expert 学習時

ハイパーパラメタ	値
per_device_train_batch_size	8
gradient_accumulation_steps	1
learning_rate	5e-5
num_train_epochs	3
r	8
lora_alpha	8
lora_dropout	0.0

表 3　追加学習時

ハイパーパラメタ	値
per_device_train_batch_size	2
gradient_accumulation_steps	4
learning_rate	5e-5
num_train_epochs	3
r	8
lora_alpha	8
lora_dropout	0.0

8）　transformers==4.35.2