修士論文

推論における重要度単語を転移させる知識蒸留法

武田 遥暉

主指導教員 白井 清昭

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 8 年 3 月

Abstract

English abstract (1200 words)
○○○○○○○○○○○○○○○○○○○○

概要

日本語の概要

○○○○○○○○○○○○○○○○○○○○

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　背景

大規模言語モデル（Large Language Model; LLM）は、大規模テキストで事前学習された言語モデルであり、文脈情報を高次元表現として獲得することでさまざまな言語タスクにおいて高い性能を示している。特に分類モデルとして用いる場合、入力文全体の意味情報を統合した表現を利用することで、文書分類や感情分析などにおいて高い性能を発揮する。

このような能力を実現するため近年の LLM は大規模なモデルサイズを有しており、その運用には多数の GPU をはじめとする計算機資源を必要とする。その結果、十分な資金力を持つ一部の組織のみが利用可能である点や、クラウド経由で GPU 資源にアクセスできない環境では導入が困難である点が、実社会への幅広い展開を妨げる課題となっている。

そのため、LLM を教師モデル、より少ないパラメータを持つモデルを生徒モデルとし、教師モデルの性能を可能な限り維持しながら、タスク遂行能力を生徒モデルへ転移する手法として知識蒸留が提案されている。特に分類タスクにおいては、性能をほぼ維持したままモデルサイズを半分程度まで削減可能であるなど、高い有効性が示されており知識蒸留はモデル軽量化の代表的な手法として確立している。

しかしながら、知識蒸留は、計算資源の制約下において教師モデルの代替として利用可能な生徒モデルを構築することを目的とした技術であるにもかかわらず、蒸留によって学習された生徒モデルは分類精度を維持できている一方、推論時に重要と判断する単語が教師モデルと一致しないことが報告されている。このような差異は、医療分野や金融分野など判断根拠の信頼性が重視される領域において致命的な問題となり得るため、生徒モデルを教師モデルの単純な代替として用いることには依然として課題が残っている。

1.2　目的

本研究では、教師モデルと同じ単語に注目して生徒モデルが推論を行うよう学習させることを目的とする。対象タスクに対して十分に学習された教師モデルを用意し、以下の手順に従って知識蒸留を適用する。

まず、教師モデルよりも小規模な未学習の生徒モデルを準備し、両モデルに同一の入力インスタンスを与えて推論を行う。次に、得られた推論結果に対してIntegrated Gradientsを用いて単語重要度を算出し、それらを重要度分布として表現する。最後に、教師モデルと生徒モデルの重要度分布が近づくよう損失関数を設計し、先行研究で提案されている知識蒸留手法に当該損失項を導入することで、生徒モデルの学習を行う。本手法により、分類精度を先行研究と同程度に維持したまま、推論時に重要と判断される単語に関して教師モデルとの整合性を向上させることを狙う。

1.3 本論文の構成

本論文の構成は以下の通りである。2章では先行研究について述べる。3章では提案手法について詳細に説明する。4章では提案手法の評価実験について述べる。最後に5章では、本研究の手法についてのまとめと今後の課題について述べる。

第2章　関連研究

本章では本研究の関連研究について述べる。本章で述べる研究トピックはKnowledge Distillation（以下、知識蒸留）[3] である。知識蒸留は、性能が高くパラメータ数も大きいモデルを教師モデルとし、パラメータ数が小さいモデルへとタスクを解くための能力を転移されるための代表的な技術である。これまでに様々な知識蒸留手法が提案されてきており、教師モデルの出力層の知識を転移するものや、隠れ状態からの知識も転移させるもの、両方から知識を転移させるもの等、様々なパターンの知識蒸留のアプローチが提案されてきた。

また、分類モデルの知識蒸留における教師モデルとしてBERT[1] を用いられることが多い。BERT を教師モデルにして、知識蒸留されたモデルの代表的なものとしてTinyBERT[4] が存在する。

これらの言語モデルの推論根拠を提供する技術としてIntegrated Gradient[5] がある。Integrated Gradients は画像処理分野で提案されたものではあるが本研究ではモデルが重要であると認識した単語を抽出するために利用する。

以下、2.1 節では、知識蒸留の原理的な設計について述べる。2.2.1 節では、教師モデルとして用いるBERT について紹介する。2.3.1 節では、知識蒸留によって構築された代表的なモデルであるTinyBERT について述べる。2.4.1 節ではモデルが推論する際の根拠を提示する技術としてIntegrated Gradients を紹介する。最後に、2.5 では、本研究の特色について述べる。

2.1　知識蒸留

知識蒸留（Knowledge Distillation）は、高性能である一方、パラメータ数が多い教師モデルから、より小規模な生徒モデルへ特定のタスクを解く能力を転移させるための代表的な技術である。一般に、教師モデルは対象タスクに対して十分に学習（ファインチューニング）されており、生徒モデルは教師モデルの振る舞いを模倣することで効率的な学習を行う。

分類タスクでは、通常、あるドメインに対して数百から数万のラベル付きインスタンスが与えられる。代表的な自然言語理解ベンチマークであるGLUE[7] は、複数の分類タスクから構成されるデータセット群であり、知識蒸留の評価においても標準的に用いられる。GLUE に含まれるMRPC（Microsoft Research Paraphrase Corpus）は、2 つの英文が意味的に等価であるかどうかを判定する二値分類タス

クである。各インスタンスにはラベル $y \in \{0, 1\}$ が付与されており、意味的に等価な場合を 1、そうでない場合を 0 と定義する。

> 例
> sentence1: Amrozi accused his brother, whom he called “the witness”, of deliberately distorting his evidence.
> sentence2: Referring to him as only “the witness”, Amrozi accused his brother of deliberately distorting his evidence.
> label: equivalent (1)

通常の分類モデルの学習では、生徒モデル f_s が入力 x に対して予測するクラス確率分布 $p_s(y \mid x)$ と、正解ラベル y との間のクロスエントロピー損失を最小化する。この損失関数は、次式で定義される。

$$\mathcal{L}_{\mathrm{CE}} = -\sum_{c} y_c \log p_s(c \mid x) \tag{2.1}$$

ここで、c はクラスを表し、y_c は正解クラスに対応するワンホット表現である。

知識蒸留では、この正解ラベルに基づく学習に加えて、同一の入力 x に対する教師モデル f_t の出力分布 $p_t(y \mid x)$ を、生徒モデルが模倣することを目的とする。教師モデルと生徒モデルの出力分布の乖離を測る指標として、Kullback–Leibler（KL）ダイバージェンスが一般に用いられる。このとき、知識蒸留損失は、次式で表される。

$$\mathcal{L}_{\mathrm{KD}} = \mathrm{KL}\big(p_t(\cdot \mid x) \,\|\, p_s(\cdot \mid x)\big) \tag{2.2}$$

ここで、$\cdot$ はクラス全体を表し、教師モデルが出力する確率分布全体を、生徒モデルが近似することを意味する。

最終的な学習では、正解ラベルに基づくクロスエントロピー損失と、教師モデルの振る舞いを模倣する知識蒸留損失を組み合わせた、次の損失関数を最小化する。

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_{\mathrm{CE}} \quad + \quad \lambda\mathcal{L}_{\mathrm{KD}} \tag{2.3}$$

ここで、$\lambda \in [0, 1]$ はハイパーパラメータであり、正解ラベルと教師モデルの知識のどちらを重視するかを制御する。

このように知識蒸留では、データセットに明示的に含まれるラベル情報だけでなく、教師モデルが獲得した予測分布という暗黙的な知識を生徒モデルに転移することで、高い性能を維持したままモデルの軽量化を実現することを目的としている。

2.2 事前学習済み言語モデル

2.2.1 BERT の事前学習

BERT[2] は、Transformer[6] を基盤とした双方向言語モデルであり、大規模コーパスを用いた事前学習によって汎用的な言語表現を獲得する。BERT の事前学習では、Masked Language Model（MLM）と Next Sentence Prediction（NSP）の 2 つのタスクが用いられる。

MLM では、入力文中の一部の単語をマスクし、その単語を周辺文脈から予測することで単語表現を学習する。MLM の損失関数は以下のように定義される。

$$\mathcal{L}_{\mathrm{MLM}} = -\sum_{i=1}^{N} \log p(t_i \mid T_i) \tag{2.4}$$

ここで、t_i はマスクされた単語、T_i は Ft_i 以外の単語からなる入力文を表す。

次に、NSP は 2 つの文が元の文書中で連続しているか否かを判定するタスクであり、文間の関係性を学習することを目的としている。NSP の損失関数は次式で表される。

$$\mathcal{L}_{\mathrm{NSP}} = -\left[y \log P_{\mathrm{NSP}} \ + \ (1-y)\log(1-P_{\mathrm{NSP}})\right] \tag{2.5}$$

ここで、y は文の連続性ラベルであり、2 つの文が連続している場合は 1、そうでない場合は 0 を取る。また、P_{NSP} は 2 文が連続しているとモデルが予測する確率を表す。BERT の事前学習では、式 (2.4) および式 (2.5) で定義される損失を同時に最小化することで、単語レベルおよび文レベルの意味関係を考慮した言語表現を獲得する。

2.2.2 教師モデルとしての BERT

本研究では教師モデルとして、BERT を用いる。BERT は事前学習によって大規模コーパスでの事前学習により汎用的な言語表現を獲得しており、このモデルに対して本研究で対象とするタスクでファインチューニングすることを行い、教師モデルとして採用する。

2.3 知識蒸留によって構築されたモデル

2.3.1 TinyBERT

TinyBERT[4] は、BERT を教師モデルとして知識蒸留を適用することで構築された軽量な言語モデルである。TinyBERT は、教師モデルの出力層だけでなく、中間層の隠れ状態や Attention 機構の重みも模倣することで、モデルサイズを大幅に削減しながら高い性能を維持することに成功している。

データ拡張 (data augmentation)

学習において TinyBERT は下流タスクでの蒸留効果を高めるためにデータ拡張を併用することが知られている。具体的には、元のデータセットに対してランダムに単語を挿入・削除・置換することで多様な入力インスタンスを生成し、頑健性を向上させる。

学習の流れ

TinyBERT の蒸留は大きく 2 段階で行われる。

事前学習段階 (pre-training)　事前学習段階では、BERT の事前学習で用いられた Masked Language Model（MLM）や Next Sentence Prediction（NSP）に加えて、中間層の隠れ状態や Attention 重みの模倣を目的とした損失を導入する。これにより、生徒モデルは教師の内部表現を効率的に学習し、下流タスクへの基盤を構築する。

下流タスクのファインチューニング (downstream fine-tuning)　下流タスクへの適応はさらに 2 段階（段階的蒸留）で実施される。

第 1 段階 — 中間層蒸留 (intermediate layer distillation) まず中間層の隠れ状態や Attention 重みの模倣損失を導入し、生徒の内部表現を教師に近づけることで下流タスクへの適応準備を行う。

第 2 段階 — 出力層蒸留 (prediction layer distillation) 続いて教師の出力（ソフトラベル）とデータのハードラベルを併用して生徒の出力層を学習させ、教師モデルの推論能力を模倣させる。

TinyBERT はこの学習の流れを通じて BERT の知識を効率的に圧縮し、軽量ながら高性能なモデルを実現している。

2.4 モデルの推論根拠の解釈に関する研究

2.4.1 Integrated Gradient

近年、深層学習モデルは画像認識や自然言語処理などの多くのタスクにおいて人間を凌駕する性能を達成している。しかし、その高い性能の代償として、モデル内部の計算プロセスは複雑化し、人間にとって理解困難な「ブラックボックス」となっていることで、モデルの推論における根拠となる特徴量を明示的に把握することが難しいという課題が存在する。

この要求に応えるための技術として、特徴量帰属法（Feature Attribution）が研究されている。特徴量帰属法は、モデルの予測結果に対する入力特徴量（例えば、画像の各ピクセルやテキストの各単語）の寄与度を定量的に算出する手法の総称である。寄与度が高い特徴量は、モデルが予測を行う上で重要な手がかりとして利用したことを示唆する。

Integrated Gradients は、ある入力 x と、情報を持たない参照点であるベースライン x' との間の経路に沿って勾配を積分することで定義される。入力ベクトル $x \in \mathbb{R}^n$ における i 番目の特徴量 x_i の寄与度 $\mathrm{IG}_i(x)$ は以下の式で計算される。

$$\mathrm{IG}_i(x) = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x - x'))}{\partial x_i} \, d\alpha \tag{2.6}$$

ここで、F は微分可能な機械学習モデル（例えばニューラルネットワークのソフトマックス出力値）、x' はベースラインベクトルを表す。α は補間係数であり、$[0, 1]$ の範囲で変化することで、ベースライン x' から入力 x への直線経路を辿る。この経路積分により、IG は以下の重要な公理的性質を満たす。

感度 (Sensitivity)　モデルの予測に変化を与える全ての特徴量は、非ゼロの寄与度を持つべきであるという性質である。従来の単純な勾配法（Simple Gradients）では、ReLU や Sigmoid などの活性化関数が飽和領域（勾配がほぼ 0 になる領域）に入ると、入力が変化しても勾配が消失し、寄与度が 0 と計算されてしまう問題があった。これを「勾配消失の問題」と呼ぶ。IG は、ベースラインから入力までの経路全体の勾配を積分するため、途中で勾配が存在すればその影響を捉えることができ、この感度の公理を満たすことができる。

完全性 (Completeness)　各次元 i における IG の総和は、常に入力 x に対するモデル出力値とベースライン x' に対するモデル出力値の差分に一致する。

$$\sum_{i=1}^{n} \mathrm{IG}_i(x) = F(x) - F(x') \tag{2.7}$$

この性質は、算出された寄与度の合計がモデルの予測の変化分を過不足なく説明することを保証するものであり、説明の信頼性を担保する上で極めて重要である。

Integrated Gradients は入力変数が連続的に変化し、微分可能であることを前提とした手法である。しかし、自然言語処理で扱うテキストデータは離散的な「単語（トークン）」の列であり、入力値を微小に変化させて勾配を求めるという操作を直接行うことができない。そこで NLP タスクにおいては、単語を連続的なベクトル表現に変換する「埋め込み層（Embedding Layer）」の値を入力とみなして IG を適用する方法が一般的に用いられる。

具体的には、入力文が T 個の単語からなり、各単語 w_t が d 次元の埋め込みベクトル $e_t \in \mathbb{R}^d$ に変換されるとする。このとき、モデルへの入力は $T \times d$ 次元の埋め込み行列 E とみなせる。これに対応するベースラインとしては、通常、全てがゼロのベクトル（ゼロ埋め込み）が用いられる。すなわち、ベースラインの埋め込みベクトルを $b_t = \mathbf{0}$ とする。

ある特定の単語 w_t の寄与度を計算するためには、まずその単語に対応する埋め込みベクトルの各次元 j についての IG 値 $\mathrm{IG}_{t,j}$ を計算する。

$$\mathrm{IG}_{t,j} = (e_{t,j} - b_{t,j}) \int_{\alpha=0}^{1} \frac{\partial F(B + \alpha(E - B))}{\partial e_{t,j}} \, d\alpha \tag{2.8}$$

ここで、E は入力文全体の埋め込み行列、B はベースライン行列である。この式は、ベースラインの埋め込みから入力の埋め込みへと徐々に特徴を変化させながら、モデルの予測値の変化に対する勾配を累積していることを意味する。

最終的に、単語 w_t としての重要度スコア S_t を得るためには、埋め込みベクトルの各次元の IG 値を集約する必要がある。集約方法としては、各次元の総和をとる方法や、L2 ノルムをとる方法などが用いられる。

このようにして算出されたスコア S_t が大きい単語ほど、モデルの予測判断に強く影響を与えた「重要な単語」であると解釈できる。

IG の計算には積分計算が必要であるが、コンピュータ上での実装においては、有限個のステップによるリーマン和近似が用いられる。積分区間 $[0, 1]$ を m 個のステップに分割した場合の近似式は以下のようになる。

$$\mathrm{IG}_i(x) \approx (x_i - x'_i) \frac{1}{m} \sum_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m}(x - x'))}{\partial x_i} \tag{2.9}$$

ここで、m は近似のステップ数である。各ステップ k においてモデルの逆伝搬計算を行う必要があるため、IG の計算コストはステップ数 m に比例して増大する。ステップ数が少なすぎると積分の近似誤差が大きくなり、積分誤差が大きくなる。一方でステップ数を多くすれば精度は向上するが、推論時間の数十倍から数百倍の計算時間を要することになる。したがって、実用上は計算コストと精度のトレードオフを考慮し、近似誤差が計算時間の許容範囲内に収まるような適切なステップ数を設定する必要がある。

2.5　本研究の特徴

従来の知識蒸留手法は、教師と生徒のモデル出力（ロジット）の確率分布を一致させることを目的としてきた。これに対し、本研究は個々の単語がモデルの最終的な予測判断にどの程度貢献したのかを示すトークン重要度分布に焦点を当てる。Integrated Gradients を用いてトークン重要度を定量化し、教師モデルと生徒モデルが同じ単語に注目して予測を行うよう学習させることで、「モデルの推論根拠」を明示的に転移させ、精度だけでなく信頼性と解釈可能性を備えた知識蒸留を実現する。

提案手法により、学習後の生徒モデルがどの単語に基づいて予測を行ったかが明らかになる。教師と生徒のトークン重要度分布を合わせることで、両者の推論プロセスがより透明になり、推論プロセスが重視される領域に対して知識蒸留モデルの代替としての信頼性を向上させる。

第3章　提案手法

3.1　概要

本研究では、GLUE に含まれるタスクを対象し、生徒モデルを学習する際に教師モデルと同じ単語に注目するように推論するための損失項を提案する。

提案手法の概要を図 3.1 に示す。本提案手法は図中の Step 1 と Step 2 の 2 段階で構成される。

Step 1 では、教師モデルと生徒モデルの双方に同一の学習データを入力し、推論結果に対して Integrated Gradients を適用することで各単語の重要度スコアを算出する。これにより、それぞれのモデルが入力文中のどの単語を推論の根拠として重要視しているかを表す重要度分布を作成する。

Step 2 では、Step 1 で得られた 2 つの重要度分布間の整合性を高めるための損失関数を定義する。この損失を最小化するように生徒モデルを学習させることで、生徒モデルは教師モデルと同様の単語に注目して推論を行うようにすることを狙う。

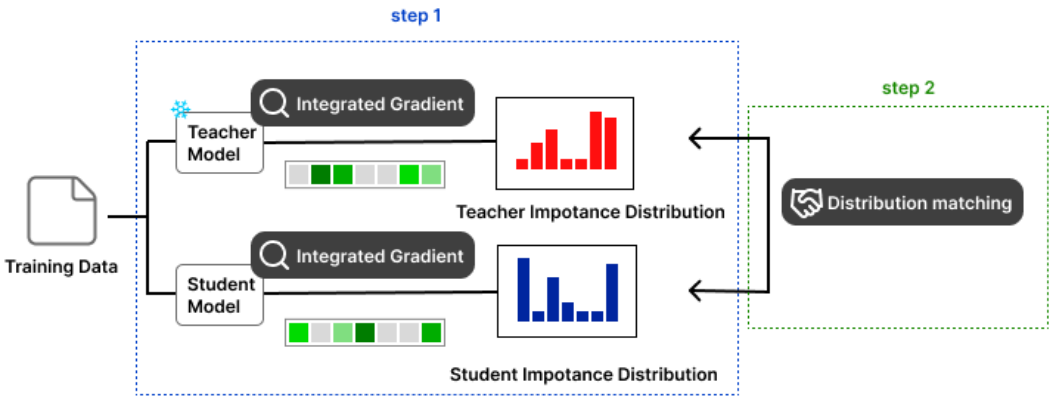

図 3.1: 提案手法の概要

本研究で提案する損失項は、提案されてきた知識蒸留手法に追加で組み込むことができる。本研究における実験については、目的とするタスクでファインチューニングされていた BERT を教師モデルとして採用し、TinyBERT の学習手法に本提案手法を組み込むことで生徒モデルの学習を行う。

3.2 重要度分布の作成

3.2.1 Integrated Gradients による単語重要度の算出

3.2.2 重要度分布の作成

3.3 重要度分布の最小化

3.4 生徒モデルの学習

3.4.1 データ拡張

3.4.2 提案手法を組み込んだ蒸留

第4章　評価

4.1　Jaccard係数による評価

4.1.1　実験条件

4.1.2　結果と考察

4.2　Rankingによる評価

4.2.1　実験条件

4.2.2　結果と考察

第5章　おわりに

5.1　本研究のまとめ

5.2　今後の課題

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[2] 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.

[3] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[4] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4163–4174, Online, November 2020. Association for Computational Linguistics.

[5] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks, 2017.

[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] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi, editors, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics.