

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

第1章	はじめに	1
1.1	背景	1
1.2	目的	1
1.3	本論文の構成	2
第2章	関連研究	3
2.1	知識蒸留	3
2.2	事前学習済み言語モデル	5
2.2.1	BERT の事前学習	5
2.2.2	教師モデルとしての BERT	5
2.3	知識蒸留によって構築されたモデル	5
2.3.1	TinyBERT	5
2.4	モデルの推論根拠の解釈に関する研究	7
2.4.1	Integrated Gradient	7
2.5	本研究の特徴	8
第3章	提案手法	10
3.1	概要	10
3.2	重要度分布の作成	11
3.2.1	Integrated Gradients による単語重要度の算出	11
3.2.2	重要度分布の作成	12
3.3	重要度分布の最小化	14
3.4	本研究で提案する最終的な蒸留損失項	14
3.5	提案手法を用いた生徒モデルの学習	16
3.5.1	ベースライン	16
3.5.2	学習タスク	16
3.5.3	損失関数の定義	18
第4章	評価	20
4.1	Jaccard 係数による評価	20
4.1.1	実験条件	20
4.1.2	結果と考察	20
4.2	Ranking による評価	20

4.2.1	実験条件	20
4.2.2	結果と考察	20
第 5 章	おわりに	21
5.1	本研究のまとめ	21
5.2	今後の課題	21

図 目 次

3.1	提案手法の概要	10
3.2	Integrated Gradients によって算出された重要度スコア	11

表 目 次

3.1	TinyBERT の GLUE での精度	16
3.2	本研究で用いる GLUE タスクの例とラベル	17
3.3	各タスクにおけるデータ拡張と学習データ量	18

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)

通常のカテゴリカルモデルの学習では、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 $p_s(y | x)$ 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。

$$\mathcal{L}_{\text{CE}} = - \sum_c y_c \log p_s(c | x) \quad (2.1)$$

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 $p_t(y | x)$ 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。

$$\mathcal{L}_{\text{KD}} = \text{KL}(p_t(\cdot | x) \| p_s(\cdot | x)) \quad (2.2)$$

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。

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

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。

$$\mathcal{L}_{\text{MLM}} = - \sum_{i=1}^N \log p(t_i | T_i) \quad (2.4)$$

ここで、 t_i はマスクされた単語、 T_i は t_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。

$$\mathcal{L}_{\text{NSP}} = -[y \log P_{\text{NSP}} + (1 - y) \log(1 - P_{\text{NSP}})] \quad (2.5)$$

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」になっていることで、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 $IG_i(x)$ は以下の式で計算される。

$$IG_i(x) = (x_i - x'_i) \int_{\alpha=0}^1 \frac{\partial F(x' + \alpha(x - x'))}{\partial x_i} d\alpha \quad (2.6)$$

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、入力に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。

$$\sum_{i=1}^n IG_i(x) = F(x) - F(x') \quad (2.7)$$

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。

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 $IG_{t,j}$ を計算する。

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

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」と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になる。

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

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。

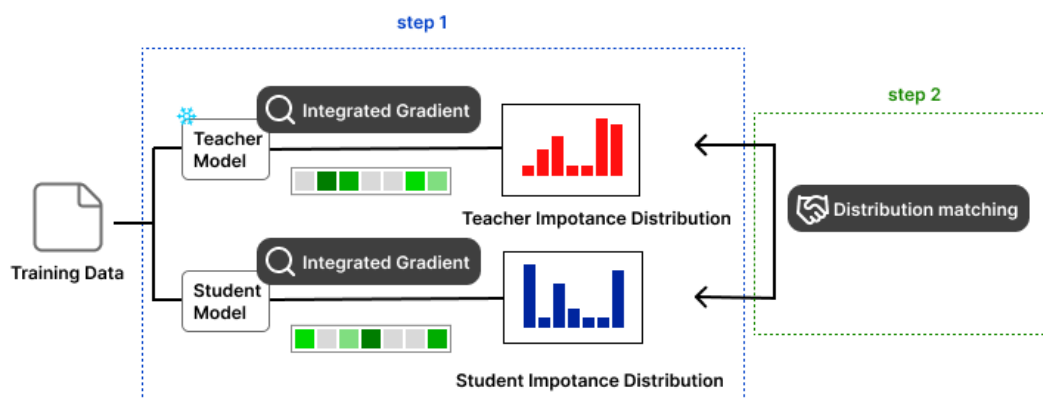


図 3.1: 提案手法の概要

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

本提案において、教師モデルと生徒モデルの重要度分布を作成する手順について述べる。具体的には学習の過程において学習データ 1 インスタンス毎に教師モデルと生徒モデルに対し Integrated Gradients を適用し、そのインスタンスのトークン長分個数の重要度スコアを算出する。GLUE/SST-2 タスクにおけるインスタンスの例を以下に示す。SST-2 タスクは、与えられた英文が肯定的か否定的かを判定する二値分類タスクである。括弧の中はモデルが予測すべきラベルに対応する。

例

sentence: seem weird and distanced

label: negative (0)

このインスタンスに対し、モデルは予測に際してトークンに分割をする。Integrated Gradients を適用する際には、トークン毎に重要度スコアが算出される。上記の例に対し、モデルが以下のようなトークン分割を行ったとする。

トークン列: [CLS] seem weird and distance ##d [SEP]

[CLS] と [SEP] は特殊トークンであり、トークナイザによって付加される。##d はサブワードである。サブワードとは、単語をさらに細かく分割したものであり、BERT のトークナイザでは未知語に対してサブワード分割が行われる。上記の例では、distanced が distance と ##d に分割されている。分割された各トークンに対し、式 (2.9) による Integrated Gradients を適用することで、各トークン毎に重要度スコアが算出される。

Integrated Gradients は埋め込みベクトルの各次元に対して計算される。最終的に、各トークンの重要度スコアは埋め込みベクトルの各次元の IG 値を集約することで得られる。具体的には、BERT の埋め込みベクトルは 768 次元であるため、各トークンに対して 768 個の IG 値が算出される。これらを集約することで、各トークンに対する 1 つの重要度スコアが得られる。

具体的に本研究で用いる SST-2 でファインチューニングされた BERT モデルに対し、SST-2 のあるインスタンスに対して Integrated Gradients を適用した際の各トークンの重要度スコアを示す。

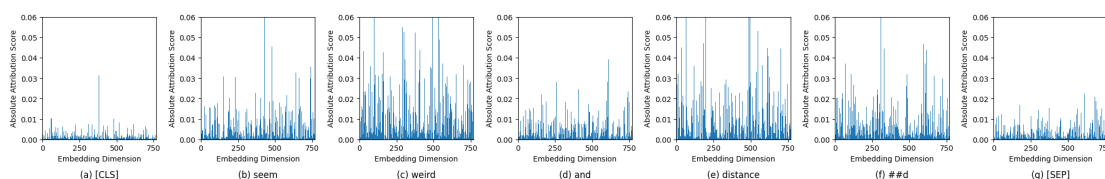


図 3.2: Integrated Gradients によって算出された重要度スコア

以上のようにして、教師モデルと生徒モデルの双方に対し、学習データ1インスタンス毎にIntegrated Gradientsを適用し、各トークンの重要度スコアを算出する。算出されたIGスコアを集約する方法については、本研究ではL2ノルムを用いる。

学習データセット内のインスタンス n における、トークン t の重要度スコア $S_{n,t}$ は、以下の式で算出される。

$$S_{n,t} = \|\mathbf{IG}_{n,t}\|_2 \quad (3.1)$$

ここで、 $\mathbf{IG}_{n,t}$ はインスタンス n のトークン t における埋め込みベクトルの各次元に対するIntegrated Gradientsの値をまとめたベクトルを表す。BERTでは埋め込みベクトルの次元数が $d = 768$ であるため、 $\mathbf{IG}_{n,t} \in \mathbb{R}^{768}$ となる。これにより、各トークンの重要度を単一のスカラー値として表現でき、トークン間の重要度比較が可能になる。本研究では、教師モデルおよび生徒モデルに対して、学習データセット内の各インスタンスごとにIntegrated Gradientsを適用し、各トークンの重要度スコアを算出する。

3.2.2 重要度分布の作成

前節で算出した各トークンの重要度スコアを用いて、教師モデルと生徒モデルの重要度分布を作成する。重要度スコア $S_{n,t}$ は各トークンがモデルの予測に与える影響の大きさを表す非負のスカラー値である。本研究では、この重要度スコアをそのまま用いるのではなく、確率分布に変換してから教師モデルと生徒モデルの比較を行う。

Integrated Gradientsによって算出される重要度スコアの値の範囲は、インスタンスごとに大きく異なる可能性がある。絶対値では大きく異なっても、各インスタンス内での相対的な重要性は類似している場合がある。しかし、重要度スコアの絶対値をそのまま用いると、この相対的な類似性を正しく捉えることができない。

この問題を解決するため、重要度スコアを確率分布に変換する。確率分布への変換では、各インスタンス内でのトークンの重要度スコアを正規化し、全トークンの確率の合計が1になるように調整する。これにより、重要度スコアの絶対的なスケールに依存せず、「各インスタンス内でどのトークンがどの程度重要か」という相対的な情報のみを表現できる。確率分布として表現することで、異なるスケールで算出された重要度スコアを統一的に扱い、教師モデルと生徒モデルの注目パターンを公平に比較できるようになる。

(1) 温度スケーリング

インスタンス n のトークン t における重要度スコア $S_{n,t}$ を温度パラメータ τ で除算する。

$$s_{n,t} = \frac{S_{n,t}}{\tau} \quad (3.2)$$

温度パラメータ τ は、重要度分布の鋭さを制御するハイパーパラメータである。 $\tau > 1$ の場合、分布は平滑化され、より均一な確率分布に近づく。これにより、重要度が中程度のトークンにも適度な確率質量が割り当てられる。一方、 $\tau < 1$ の場合、分布は鋭くなり、重要度が高いトークンへの確率集中が強調される。 $\tau = 1$ の場合は、スケーリングを行わず、元の重要度スコアをそのまま用いることに相当する。本研究では、教師モデルと生徒モデルの重要度分布の比較において、いくつかの設定を試した結果、タスクによって温度を調整することが有効であると判断した。

(2) ソフトマックス正規化

次に、スケーリング後のスコアをソフトマックス関数で正規化し、確率分布に変換する。教師モデルの重要度分布 $p_{n,t}^T$ は以下の式で得られる：

$$p_{n,t}^T = \frac{\exp(s_{n,t}^T)}{\sum_{t'=1}^L \exp(s_{n,t'}^T)} \quad (3.3)$$

同様に、生徒モデルの重要度分布 $p_{n,t}^S$ は以下の式で得られる：

$$p_{n,t}^S = \frac{\exp(s_{n,t}^S)}{\sum_{t'=1}^L \exp(s_{n,t'}^S)} \quad (3.4)$$

ここで、 $s_{n,t}^T = S_{n,t}^T/\tau$ は教師モデルの温度スケーリング後のスコア、 $s_{n,t}^S = S_{n,t}^S/\tau$ は生徒モデルの温度スケーリング後のスコアを表す。 L はインスタンス n のトークン長を表す。この正規化により、 $\sum_{t=1}^L p_{n,t}^T = 1$ かつ $\sum_{t=1}^L p_{n,t}^S = 1$ を満たす確率分布が得られ、教師モデルと生徒モデルの重要度分布を比較できるようになる。

3.3 重要度分布の最小化

前節で作成した教師モデルと生徒モデルの重要度分布の類似度を測定し、その類似度を最大化するように生徒モデルを学習させる。本研究では、2つの分布の類似度を測る指標として Jaccard 係数を用いる。Jaccard 係数は、2つの集合の類似度を測る指標であり、2つの集合の共通部分の大きさを和集合の大きさで割った値として定義される：

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (3.5)$$

ここで、 A と B はそれぞれ2つの集合を表す。Jaccard 係数は 0 から 1 の値をとり、2つの集合が完全に一致する場合に 1、共通部分が存在しない場合に 0 となる。

しかしながら、従来の Jaccard 係数は離散的な集合に対して定義される指標である。本研究では、生徒モデルの学習において、損失関数を最小化するようにパラメータを更新する必要がある、そのためには損失関数が生徒モデルのパラメータに関して微分可能でなければならない。従来の Jaccard 係数は微分不可能であるため、勾配降下法によるパラメータ更新に用いることができない。

そこで、微分可能な Soft Jaccard Similarity[?] を用いる。Soft Jaccard Similarity は、従来の Jaccard 係数を連続的な確率分布に拡張したものであり、以下のように定義される。

$$J_{\text{soft}}(\mathbf{p}^T, \mathbf{p}^S) = \frac{\sum_{t=1}^L p_{n,t}^T \cdot p_{n,t}^S}{\sum_{t=1}^L (p_{n,t}^T)^2 + \sum_{t=1}^L (p_{n,t}^S)^2 - \sum_{t=1}^L p_{n,t}^T \cdot p_{n,t}^S} \quad (3.6)$$

ここで、 \mathbf{p}^T は教師モデルの重要度分布、 \mathbf{p}^S は生徒モデルの重要度分布を表す。分子は2つの分布の内積であり、従来の Jaccard 係数における積集合に相当する。分母は2つの分布の自乗和の和から内積を引いたものであり、和集合に相当する。Soft Jaccard Similarity は確率分布に対して連続的に定義されるため微分可能であり、損失関数として用いて勾配降下法により生徒モデルのパラメータを更新することが可能になる。

3.4 本研究で提案する最終的な蒸留損失項

本研究では、今までに述べた重要度分布の類似度を最大化する損失項を、知識蒸留における生徒モデルの最終的な損失関数に追加する。具体的には、以下のよう

$$\mathcal{L}_{\text{SJ}} = 1 - J_{\text{soft}}(\mathbf{p}^T, \mathbf{p}^S) \quad (3.7)$$

ここで、 \mathcal{L}_{SJ} は本研究で提案する蒸留損失項を表す。 $J_{\text{soft}}(\mathbf{p}^T, \mathbf{p}^S)$ は式 (3.6) で定義される Soft Jaccard Similarity であり、教師モデルと生徒モデルの重要度分布の類似度を $[0, 1]$ の範囲で評価する指標である。値が大きいほど両者の注目パターンが一致していることを意味する。

本研究では、類似度の最大化を目的とする代わりに、最小化問題として扱うために損失項を $1 - J_{\text{soft}}$ の形で定義している。これにより、完全一致の場合は $\mathcal{L}_{\text{SJ}} = 0$ となり、類似性が低い場合ほど損失が大きくなるという直観的な対応が得られる。本研究で提案は他の知識蒸留手法に組み込むことを想定しているため、交差エントロピー損失や既存の損失関数に加算する形で用いるため、最小化問題として扱うことは、他の損失関数と整合的に組み合わせることができる。

3.5 提案手法を用いた生徒モデルの学習

本章では、提案する蒸留損失項を用いた生徒モデルの学習方法について述べる。本研究においては、GLUE の各タスクにおいてファインチューニングされた BERT を教師として知識蒸留された TinyBERT に対して、提案手法を適用する。

3.5.1 ベースライン

ベースラインとして利用する TinyBERT はファインチューニングについて 2 段階の蒸留を行う。第 1 段階では中間層の隠れ状態について生徒の内部表現を教師に近づける。第 2 段階では教師の出力と学習データのラベルを併用して生徒の出力層を学習させる。本研究では Encoder ブロックを 6 層および 4 層の 2 種類を採用する。精度を表 3.1 に示す。

表 3.1: TinyBERT の GLUE での精度

Model	MRPC	CoLA	STS-B	RTE	QQP	MNLI	QNLI	SST-2
Metric	acc	mcc	pearson	acc	acc	acc (m/mm)	acc	acc
BERT-base (FineTuned)	88.24	53.39	87.95	72.92	90.88	83.9/83.4	91.45	92.43
TinyBERT-6L	86.27	48.72	87.70	63.90	90.87	83.94	90.33	91.74
TinyBERT-4L	85.54	37.48	87.41	63.18	90.12	81.80	87.06	90.94

3.5.2 学習タスク

本研究では、GLUE タスクに含まれるデータを対象とする。GLUE は自然言語理解の多様な側面を評価するために設計されたベンチマークであり、具体的なタスクの詳細を述べる。

CoLA 文の文法的妥当性判定。与えられた文が文法的に自然 (acceptable) か否かを判定する二値分類タスク

SST-2 文の感情判定 (感情分析)。短いレビュー文が肯定的か否定的かを判定する二値分類タスク

MRPC パラフレーズ判定。2つの文が意味的に同義 (paraphrase) かどうかを判定する二値分類タスク

STS-B 文対の意味的類似度評価。2文の意味的類似度を連続値 (0-5) で評価する回帰タスク

QQP 質問ペアの重複判定。Quora 上の 2つの質問が同じ意図・内容かを判定する二値分類タスク

MNLI 自然言語推論 (Multi-Genre NLI)。前提と仮説の関係が含意 (entailment)、矛盾 (contradiction)、中立 (neutral) のいずれかを判定する三値分類タスク

QNLI 質問と文の関連判定。与えられた文が質問に対する答えや関連情報を含むかどうかを判定する二値分類タスク

RTE テキスト含意判定タスク。前提と仮説の含意関係 (含意／非含意) を判定する二値分類タスク

表 3.2: 本研究で用いる GLUE タスクの例とラベル

タスク	入力例	ラベル
CoLA	“He go to the store.”	unacceptable
SST-2	“The movie was fantastic and deeply moving.”	positive
MRPC	“The company reported profits.” “The firm announced earnings.”	equivalent
STS-B	“A man is playing a guitar.” “Someone plays a stringed instrument.”	4.5
QQP	“How to bake a cake?” “What is the process to make a cake?”	duplicate
MNLI	前提: “A dog is sleeping on the couch.” 仮説: “An animal is resting on furniture.”	entailment
QNLI	質問: “Who wrote Hamlet?” 文: “Hamlet was written by Shakespeare.”	relevant / entailment
RTE	前提: “The sky is clear and blue.” 仮説: “The sky has no color.”	not-entailed

また、本研究においては TinyBERT の公式設定に従い、各タスクに関してデータ拡張を用いる。学習においては拡張されたデータを用いるが、一部のタスクに関して本提案の適用により計算コストが増大するため、学習データのランダムサンプリングにより拡張データから一定割合のデータを選択して用いる。各タスクにおける拡張データの利用状況を表 3.3 に示す。

表 3.3: 各タスクにおけるデータ拡張と学習データ量

	オリジナル学習データ量	ランダムサンプリングの有無	学習データ量
MRPC	3,669	無	225,098
CoLA	8,552	無	211,059
STS-B	5,750	無	320,113
RTE	2,491	無	68,796
QQP	363,847	有	1,221,763
MNLI	392,703	有	1,091,538
QNLI	104,744	無	1,250,571
SST-2	67,350	無	1,107,510

学習データに関して、TinyBERTT での公式拡張データ量は QQP が 7,572,066 件、MNLI が 8,094,053 件に拡張されている。拡張は、オリジナルの学習データからある単語をランダムにマスクし、BERT で予測した単語で置換することで行われる。タスクによって拡張率は異なるが、オリジナルの学習データ 1 インスタンスにつき複数の拡張データが生成されるため、拡張後のデータ量は大幅に増加する。QQP と MNLI に関しては他のタスクよりも圧倒的にデータ量が増大しており、本研究の計算量では膨大な時間を要した。よって、QQP と MNLI に関しては、同じオリジナルの学習データから生成された拡張データ群からランダムに一定量のデータ数を均等にランダムサンプリングすることで適切な学習時間とデータの偏りを無くした。

3.5.3 損失関数の定義

本研究の実験では、TinyBERT の実装をベースに提案手法である蒸留項を導入し、蒸留をすることで TinyBERT の精度を維持しつつ、教師モデルと同じ単語に着目して推論を行うことを目指す。TinyBERT の学習は 2 段階で行われ、第 1 段階で中間層の隠れ状態や Attention 重みを教師モデルから模倣したモデルを用意する。本研究では、第 2 段階で出力層を教師から模倣する段階に介入する。

第 2 段階では、予測層蒸留損失 $\mathcal{L}_{\text{pred}}$ が用いられる。 $\mathcal{L}_{\text{pred}}$ は、教師モデルのロジット（ソフトラベル）と生徒モデルのロジットの間の交差エントロピーであり、次式で定義される。

$$\mathcal{L}_{\text{pred}} = \text{CE}(\mathbf{z}^T/t, \mathbf{z}^S/t) \quad (3.8)$$

ここで、 \mathbf{z}^S と \mathbf{z}^T はそれぞれ生徒モデルと教師モデルが予測するロジット、CE はクロスエントロピー損失、 t は温度パラメータを表す。TinyBERT の実験では $t = 1$ が有効であると報告されており、本研究でも同様に設定する。

本研究では、提案する蒸留損失項を TinyBERT の予測層蒸留損失に加算する形で最終的な損失関数を定義する。最終的な損失関数は以下のように表される。

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda \mathcal{L}_{\text{SJ}} \quad (3.9)$$

ここで、 λ は提案する蒸留損失項の重みを制御するハイパーパラメータである。 λ の値を調整することで、予測精度と注目パターン的一致度のトレードオフを制御できる。本研究では、各タスクにおいて λ の値をいくつか試し、タスクによって最適な値を選択した。

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)*, pages 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*, pages 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 Chrupala, and Afra Alishahi, editors, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics.