

日本語論文の構造を利用した抽象型自動要約に関する 研究

電気通信大学大学院 情報理工学研究科
博士(工学) 学位申請論文
西野研究室所属
沢畑英之

2024年3月25日

日本語論文の構造を利用した抽象型自動要約に関する研究

博士論文審査委員会

主査 西野哲郎 教授

副査 庄野逸 教授

大須賀昭彦 教授

柏原昭博 教授

内海彰 教授

著作権所有者
沢畑英之 2024 年

Abstract

Title: Research on Automatic Abstractive Summarization of Paper Using the Structure of Japanese Papers.

Summarizing is the process of rewriting a book or document so that its contents are expressed in a concise manner. However, in order to perform a summary, it is necessary to accurately grasp the contents of the document, extract the important parts of the contents, and summarize them. In addition, depending on the type of document, it may be required to summarize only specific parts of the document instead of the whole document, which requires a lot of labor to be done manually.

For this reason, research on automatic summarization is a field in demand, and there have been many studies on automatic summarization, both in English and Japanese. However, most of these studies have focused on short documents such as news articles, and the number of studies on automatic summarization of long documents written in Japanese is small.

Therefore, in this study, we propose an automatic summarization method for long documents, i.e., articles written in Japanese. The proposed method automatically selects sections for summarization from the title names of the sections, generates a summary for each selected clause, and combines them to form a summary of the paper.

Comparing the proposed method with the case where the summary is generated using the full text, we shown that the proposed method performs better than the case where the summary is generated using the full text. We also compared our method with chatGPT, which is a large general-purpose language model, and showed that our proposed method outperforms it. The results shown that our proposed method is an effective method for summarizing papers. Our proposed method can be applied not only for generating summaries of articles, but also for assisting in writing and reading abstracts of articles, and for applying to documents with similar structures.

概要

要約とは、ある文書について、その文書の重要な内容を過不足なく表現することである。要約は様々なところで用いられており、例えば新聞記事の見出しや、小説、評論などの本の裏表紙のあらすじ、ニュースポータルサイトの3行要約など様々なところで使われている。しかし、要約は非常に労力のかかるものであるため、自動化が望まれている分野でもある。そのために、現在では自動要約研究は盛んであり、多くの研究がなされてきている。ただし、これらの多くはニュース記事のような短文を対象とした研究であり、長文の文書を対象とした研究は数が少なく、特に日本語で記述された長文文書を対象とした研究の数は非常に少ない。この現状の主な理由は2つある。1つ目は、長文文書に対する自動要約では要約を生成するための計算量が、短文の記事と比較して、増加することであり、2つ目は学習に用いることのできるデータセットの数が非常に少ないことである。そこで本研究では、長文文書として、日本語で記述された論文を対象にした自動要約手法を提案する。提案手法では、これらの問題点に対応するため、節ごとに自動要約を行う。節ごとに要約を行うことで、1回の要約生成における計算量を抑え、学習データ数を増加させることが可能となり、少ない件数のデータセットでも性能を改善することが可能となることを示す。加えて、提案手法ではより要約の性能を向上させるために、要約する節を選択する。論文の節同士の関係から、要約において必要となる節を選択することで、冗長性の少ない要約を生成できることが期待できる。

本論文は次の構成からなる。第1章では、序論として研究背景、及び目的について述べる。第2章、第3章では自動要約手法とそれに対する評価手法など、諸定義について述べ、本研究の提案手法の技術的背景を明確にする。

第4章では、日本語における自動要約手法の性能の検証について述べる。第2章で紹介される自動要約技術の多くは、英語でのみ性能が評価されており、日本語における性能は未知数である。そこで本章では、英語以外でも性能が発揮されることが示唆されているトピックベースの抽出型要約と、多言語事前学習モデル mT5 を用いた抽象型要約について、日本語データセットを対象に、自動要約性能を評価した。評価の結果、トピックベースの抽出型要約では英語と同様の性能を発揮し、mT5 を用いた抽象型要約では日本語ニュース記事データセットによるファインチューニングによって性能が改善されたことが確認された。このことから、トピックベースの抽出型要

約と mT5 による抽象型要約は日本語における自動要約でも有効であることが確認された。

第 5 章では、論文を対象にした要約にて、要約する節を限定する自動要約手法について提案する。論文の構造と要旨の構造の関係から、まず、序論の節と結論の節に限定した抽出型要約手法について提案する。序論及び、結論は要旨には、それらの内容が含まれているため、全文から抽出型要約を行う場合よりも、序論と結論から生成することで、冗長性の少ない要約を得られることが期待される。この手法に対して、全文から抽出型要約を行った場合と比較を行った。結果、序論と結論からの要約の方が評価は高く、要約の対象となる節を限定する手法は有効であることが確認できた。そして、この結果から、提案手法を、序論と結論以外の、他の節の内容を要約に含めるために要約に使用する節を自動で選択し、自動要約を行う手法に発展させる。序論と結論以外の節については、著者ごとに独自の節のタイトル名を付ける傾向にあること、要旨に序論と結論以外の内容をどれだけ含めるかは論文ごとに異なるため、キーワード一致などのヒューリスティックな方法では選択しきれない場合がある。そこで、節のタイトル名の一覧から、要約に用いる節を自動で選択するモデルを用いた抽象型要約手法を提案する。全文入力による抽象型要約と比較した結果、提案手法による要約が有効であることが確認できた。さらに、序論と結論のみに要約を限定した場合と比較しても、節選択モデルを用いた抽象型要約のほうが評価が高いことも確認できた。

第 6 章では、汎用言語モデルである GPT を用いた対話型 AI ツール chatGPT との比較を行う。chatGPT は大型事前学習モデルである GPT に Reinforcement Learning from Human feedback (RLHF) を適用した手法で、人間からの評価を報酬とした強化学習により、人間にとって好ましい出力を行うことができるようになっており、要約生成を含んだ様々なタスクを行うことが可能である。この章では、chatGPT による論文要約と第 5 章にて提案した節選択による抽象型要約手法による論文要約を比較した。

結果、本研究の提案手法の評価が chatGPT より高いことが確認できた。このことより、本研究の提案手法は論文要約という点において chatGPT よりも性能が高いことが示された。第 7 章では、これまでの実験の結果についてまとめ、今後の課題と発展について考察する。本研究は、現状、数少ない日本語を対象とした論文要約において、研究の数が少ない要因である学習データ数の不足と、要約生成時の計算量の増加について、これらの問題点を解決する手法を提案し、論文以外の類似した構造の文書でも適用できる応用性と、汎用言語モデルである GPT を用いた AI ツールである chatGPT に対して日本語論文自動要約において性能が高いことを示した。

目次

第 1 章	序論	10
1.1	研究背景	10
1.2	研究目的	12
1.3	本論文の構成	13
第 2 章	自動要約について	15
2.1	構成	15
2.2	自動要約の種類	15
2.3	抽出型要約	17
2.3.1	抽出型要約手法の変遷	17
2.3.2	特徴量を用いる手法	18
2.3.3	数理的な手法	19
2.3.4	BERT を用いる手法	25
2.4	抽象型要約	26
2.4.1	抽象型要約の台頭	26
2.4.2	RNN	26
2.4.3	Encoder-Decoder モデル	30
2.4.4	Pointer-generator Network	33
2.4.5	Discourse-aware Attention Model	35
2.4.6	Transformer	38
2.4.7	Transformer の発展モデル	41
第 3 章	自動要約に対する評価手法	47
3.1	構成	47
3.2	要約の評価と自動化	47
3.3	ROUGE	49
3.3.1	ROUGE-N	49

3.3.2	ROUGE-L	50
3.3.3	ROUGE-S	53
3.3.4	ROUGE の問題点	54
3.4	BERTScore	55
3.4.1	BERTScore による評価	55
3.5	QAEval	56
3.5.1	概要	56
3.5.2	名詞句の抽出	57
3.5.3	質問の生成と回答	57
3.5.4	回答に対する評価	58
3.5.5	要約評価の指標について	58
第 4 章	日本語における自動要約性能の評価	60
4.1	本章について	60
4.2	LSA による抽出型要約手法の評価	60
4.2.1	使用コーパス	61
4.2.2	結果と考察	62
4.3	LDA への置換に伴う性能の変化の評価	62
4.3.1	概要	62
4.3.2	結果と考察	63
4.4	mT5 による抽象型要約手法の評価	63
4.4.1	使用コーパス	63
4.4.2	結果と考察	64
4.5	本章のまとめ	64
第 5 章	文章構造に着目した論文要約手法	66
5.1	既存の論文の自動要約手法について	66
5.2	論文と要旨の構造	67
5.2.1	論文の構造	67
5.2.2	要旨の構造	68
5.3	序論と結論からの要約生成による性能改善	69
5.3.1	実験	70
5.3.2	結果と考察	71
5.4	節選択モデルによる論文要約手法	72
5.4.1	節のタイトルと要旨の構造の関係	72

5.4.2	提案手法の概要	74
5.4.3	要約モデル	76
5.4.4	節選択モデル	76
5.4.5	全文からの要約生成との比較	78
5.4.6	実験結果と考察	80
5.5	本章のまとめ	82
第 6 章	汎用言語モデルと提案手法の性能比較	83
6.1	背景	83
6.2	chatGPT	83
6.3	InstructGPT	84
6.3.1	SFT モデル	86
6.3.2	報酬モデル	86
6.3.3	SFT モデルの再学習	87
6.3.4	InstructGPT と chatGPT の違い	88
6.4	chatGPT との論文要約性能の比較	88
6.4.1	概要	88
6.4.2	結果と考察	89
6.5	本章のまとめ	90
第 7 章	まとめ	92
7.1	構成	92
7.2	本研究のまとめ	92
7.3	提案手法に対する考察と課題	93
7.3.1	要約前の節の選択	93
7.3.2	要約生成モデルと学習データ	94
7.3.3	他言語への提案手法の適用について	95
7.3.4	他分野論文への応用	96
7.4	提案手法の応用	96
7.4.1	提案手法の要約分野への応用について	96
7.4.2	要約分野以外への応用	97
関連論文		112
参考文献		113

第 1 章

序論

1.1 研究背景

要約とは、ある文書の内容について、その文書の重要な内容を過不足なく表現することである。要約は様々なところで用いられており、例えば新聞記事の見出しや、小説、評論などの本の裏表紙のあらすじなど様々なところで使われている。他にも、ニュースサイトにおいてはニュース記事の要約を事前に掲載するなど、現在においても活用がされている技術である。

しかし、要約は非常に労力を要する技術でもある。その主な理由は2つある。1つ目は要約を行うには文書の内容を十分に理解する必要があるという点である。理解するには文書を一度完全に読んだのちに、さらに内容への理解を深めるために二度、三度と読み直す必要もある。2つ目の理由は重要な内容の選択と抽出が難しいという点である。文書を読み込み、内容を理解した後は、次に要約に必要な重要な内容を含む部分を文書内から抽出しなければならない。文書によって重要な内容というのは異なるため、内容を理解して文書ごとに重要な内容というのを見つけ出して要約を作っていく必要がある。また、文書の種類においては作成する要約の内容についても注意を払う必要がある。例えば、小説であるならば物語の内容をばらさない様を書く必要があり、より多くの労力を要することになる。

このような背景があるため、自動要約研究は盛んである。例えば、文書内の単語の種類や文の長さなどの特徴量を利用して要約を生成する研究 [1] や、ニューラルネットワークを用いて自動要約を行う研究 [2] もある。これらの多くは英語の文書を対象として行われた研究であるが、昨今では日本語を対象とした自動要約研究もされてきており、さらには日本語用のコーパスも公開されるなどより自動要約の研究が進んでいる。さらには企業が自動要約モデルによるプロダクトを公開し、実際に使用している。例えば朝日新聞社では自動要約モデルを用いた API, TSUNA[3] を公開しており、新聞見出しの生成から、新幹線などの電光掲示板において表示される速報の生成にも活用されている。NTT では COTOHA Summarize[4] という API を公開しており、こちらは通常の要約に加えて、会話内容や講演といった音声の内容を文字に起こし、それらの内容を要

約するという機能もある。

このように、現状、自動要約研究というのは英語のみならず、日本語でも研究が多くなされてきており、そして研究結果を実際のプロダクトで使用する場面が見られるようになってきた。しかし、現状の自動要約研究における研究の対象の多くはニュース記事といった、比較的短めの文書であることが多いのに対して、小説、論文のような長文の文書に対する自動要約研究というのは数が少なく、特に日本語で書かれた長文の文書を対象として行われた自動要約研究は希少である。日本語で書かれた長文文書の自動要約の研究が希少である主な理由は、文書が長いほど自動要約を行うのに必要な計算量が増えていくためである。多くの要約モデルでは、1回の自動要約において、入力される要約対象の文書の長さが長いほど、特徴量の計算や、ニューラルネットワークであれば誤差逆伝播するノードの数が増えるなど、計算量はどのような場面においても増加してしまう。また、要約モデルにおける正常な学習が可能な単語数は多くなく、おおよそ512単語ほどとされている。そのため、一般的な要約モデルでは、要約の対象はニュース記事などの短文の文書が主となっている。対して、例えば論文を要約の対象として行う場合、1つの文書あたり1000単語は優に超えていき、また要約も同様に長いため、モデルに学習を行わせても、正常に学習できないという問題がある。そのため、自動要約モデルの研究では計算量の増えにくい、短めの文からなる文書であるニュース記事で構成されるコーパスを対象に行われることが多く、長文を対象とした要約手法の研究の数が非常に少ない。現実における文書というのは、ニュース記事だけでなく、先にあげた論文や小説以外にも多種多様な文書が存在しており、長さも同様にさまざまである。そのため、現状の自動要約手法のみでは、現実にある文書やテキストに対して適応可能な範囲が非常に少ない。

故に、長文に対する自動要約手法を確立することで、ニュース記事以外の他種類の文書やテキストにも自動要約を行うことが今後可能になることが推測される。さらに先述したように現在においては日本語を対象とした要約モデル自体は存在するが、日本語を対象とした長文自動要約は数少なく、日本語を含んだより多くの文書がデジタル化していく現在においては日本語に特化した長文要約モデルは必須である。事実、ニュース記事をはじめとして、現在では小説や説明書などより多くの文書がデジタル化し、インターネット上で公開されているため、多言語モデルではカバーできない表現を含んだ日本語で書かれた文書、例えば大正や明治時代に書かれたような文語体を含む文書においても自動要約を行うには、日本語に特化したモデルというのは必須である。とはいえ、現在において日本語のみで構築されているコーパスは非常に種類に乏しく、その多くはニュース記事のみで構築されており、長文から構築されているコーパスは数が少ない状態である。そのため、ただ単に長文に対応できるだけの日本語の要約生成モデルを構築するだけではなく、少ない学習データでも構築することが可能な、日本語に特化した要約生成モデルの構築手法が、日本語特化の長文文書の要約生成研究において必要である。

1.2 研究目的

本研究では、このような背景から、長文文書要約手法として、日本語で記述された論文を対象とした要約生成手法を提案する。長文の自動要約における要約の対象として、論文が採用される理由として以下が挙げられる。

- 論文著者の書いた要約が本文に含まれており、学習及び評価がしやすい。
- 構造が明確であるため、文書の構造を利用しやすい。

特に後者が重要な点である。ニュース記事は構造が不明瞭であるが、論文は節などの、明確な構造が存在しており、入力に構造の情報を含ませたり、要約モデルなどに構造を落とし込むことができる。英語で書かれた論文を対象とした長文の自動要約モデルとして Discourse-Aware Attention モデル [5] がある。このモデルでは、節ごとに特徴量に変換して要約文の生成に用いるため、節同士の関係性を利用することが可能である。

しかし、現在モデルの学習用として公開されている論文から構成されるコーパスの数は少なく、英語であれば arXiv datasets 及び PubMed datasets[5]、日本語であれば自然言語処理学会論文誌コーパス [6] のみである。さらに収録されている件数も少なく、一般的な要約データセットである NYTimes datasets は 61 万 5 千件であるのに対して、arXiv datasets は 21 万 5 千件、PubMed datasets は 13 万 3 千件であり、自然言語処理学会論文誌コーパスは凡そ 692 件程度と非常に少ないという問題点もある。この問題点に対しては、自動要約モデルの学習手法として DANCER[7] が提案されている。この手法の詳細は後の章にて述べるが、簡潔に述べると、文書内のセクションごとに学習を行うというもので、これにより、数の少ないコーパスにおいても学習データ数を増やすことが可能となり、要約モデルの性能改善につながる。

このように、論文に対する自動要約研究においては論文自体の構造を利用しているものが多い。そこで、本研究では論文自体の構造だけでなく、論文と要旨の構造の関係性に着目する。論文における要旨というのは、論文内の序論や結論、提案手法、実験などの節の要約をまとめたものである。つまり、これらに対応する節のみを抽出し、要約の生成に使用することで全文を入力する場合よりも、冗長性の少ない要約を生成することが可能になると考えられる。また、節のタイトルというのは対応する節の内容をまとめたものであり、どのような内容が書かれているのかを端的に表しているものであると考えられる。そのため、節のタイトル名から該当の節の内容が要旨に含まれるかどうかの判断をすることは可能であると推測できる。

しかし、日本語で書かれている論文の多くは、独自の節タイトルをつける傾向にあるため、序論や結論にあたる節はともかく、手法などの節はキーワード抽出などのヒューリスティックな方法で必要なものを選択するのは困難である。そこで、本研究ではニューラルネットワークを用いて、要旨に含まれやすい節のタイトルを学習した節選択モデルを構築する。構築の際には、一つの節

タイトルから選択を行うのではなく、論文に含まれるすべての節のタイトルを用いて選択を行うようにする。これにより、タイトル名だけでなく、他のタイトル名の情報を用いることができるようになり、選択の精度が上がり、より良い要約の生成を行うことが期待できる。

本研究では、この選択モデルを用いて要約に必要な節を選択し、選択された節それぞれから要約を要約モデルにて生成する手法を提案する。節ごとに要約を行うことで、全文入力を行った場合と比較して1回あたりの要約における計算量が減少できることが期待できる。また、節ごとに要約を行うため、上述した DANCER を学習に適用することが可能となり、要約モデル構築時の学習データ数を増加させることができる。さらに、要約に必要な節を事前に選択しておくことで、先述したように冗長性の少ない要約の生成が可能となることが期待できる。アルゴリズム 1 に簡単な本研究の提案手法の流れを示す。

Algorithm 1: 提案手法のアルゴリズム。事前に節選択モデルにて要約に使用する節を選び、それらを要約モデルに入力して生成する。

Input : $sections$ = 論文内の各節

$names$ = 論文内の節タイトル

Output: $summary$ = 論文の要約

$t_1, \dots, t_N \leftarrow Model_{sec}(names)$ if $t_1, \dots, t_N \in sections$; // 節選択モデルで要約対象の節を選択

$s_1, \dots, s_N \leftarrow Model_{sum}(t_1, \dots, t_N)$; // 選択された節ごとに要約を生成

$summary \leftarrow Concat(s_1, \dots, s_N)$; // 要約同士を結合して、論文の要約とする。

本研究では、まず提案手法についての有効性を検証する前に、既存の自動要約手法の日本語における性能を評価し、日本語でも英語と同程度に動作するのかどうかを検証した。次に、日本語でも機能する自動要約手法を用いて、論文と要旨の構造に着目した提案手法が有効なのかどうか、全文入力と比較して検証した。最後に汎用言語モデルと比較を行い、論文要約において本研究の提案手法が有効かどうかを評価した。まとめると、本研究の目的は、既存の自動要約手法の中から日本語において有効である手法を用いて、論文と要旨の構造の関係性を利用した日本語論文要約性能の有効性を示すことである。

1.3 本論文の構成

本論文は第2章及び第3章で自動要約手法と評価手法について述べ、本研究における技術的背景を示す。第4章では日本語における抽出型要約と抽象型要約の性能について評価し、日本語においても英語と同程度に動作するのかが評価する。第5章では本研究の提案手法について説明し、実際に有効かどうかを検証について述べる。第6章では chatGPT との比較検証について述べ、

第 7 章で本研究のまとめ, 提案手法についての考察と今後の課題と展望, そして本研究の提案手法の要約分野とそれ以外の分野での応用についての考察を述べる.

第2章

自動要約について

2.1 構成

本章は次の内容で構成される。

- 自動要約の種類
- 抽出型要約について
- 抽象型要約について
- 論文に対する自動要約研究の紹介

まず、自動要約の種類として、どのような種類があり、その長所と短所について述べる。次に抽出型要約について述べる。この節では、抽出型要約手法についてその計算方法、及び生成例を交えて解説する。そして、抽象型要約について述べる。現在の抽象型要約における基本であるニューラルネットワークの基礎について解説し、自動要約に用いられるニューラルネットワークについて解説する。最後に論文の自動要約について、先行研究を紹介する。どのような方法で実現しているのか、その手法について解説する。

2.2 自動要約の種類

自動要約研究において、様々な観点から分類することができる。例えば、要約を行う対象の文書が単一か複数か、全体の情報を要約するか、それとも特定のキーワードに基づいて要約するか、などの観点がある。一例として自動要約の分類を表 2.1 に示す。自動要約研究では主に、要約の生成の仕方に着目し、抽出型要約か抽象型要約かどうかの分類を行う。表 2.1 で述べている通り、抽出型要約は要約対象の文書の本文から要約となる文を選択し、それを要約の本文とする手法である。対して、抽象型要約は対象の文書の情報を元にゼロから要約文を生成する手法のことを指す。これらの手法の違いを示したイメージ図を図 2.1 に示す。また、それぞれの手法の長所と短所を以

観点	自動要約の種類	概要
入力する文書の数	単一文書要約	1つの文書から要約を行う
	複数文書要約	複数の文書からそれらの内容をまとめた要約を生成
要約に対する指示	全体要約	対象の文書そのものについて要約
	キーワード要約	指定されたキーワードについて要約を生成
	アスペクトベース要約	特定の事象について要約を生成
要約の生成の仕方	抽出型要約	対象の文書の本文から要約文として抽出して生成
	抽象型要約	文書の情報からゼロから要約を生成

表 2.1 自動要約の種別の一覧。他にも特定の分野に特化した要約や、言語に特化した要約など様々な手法が存在する。

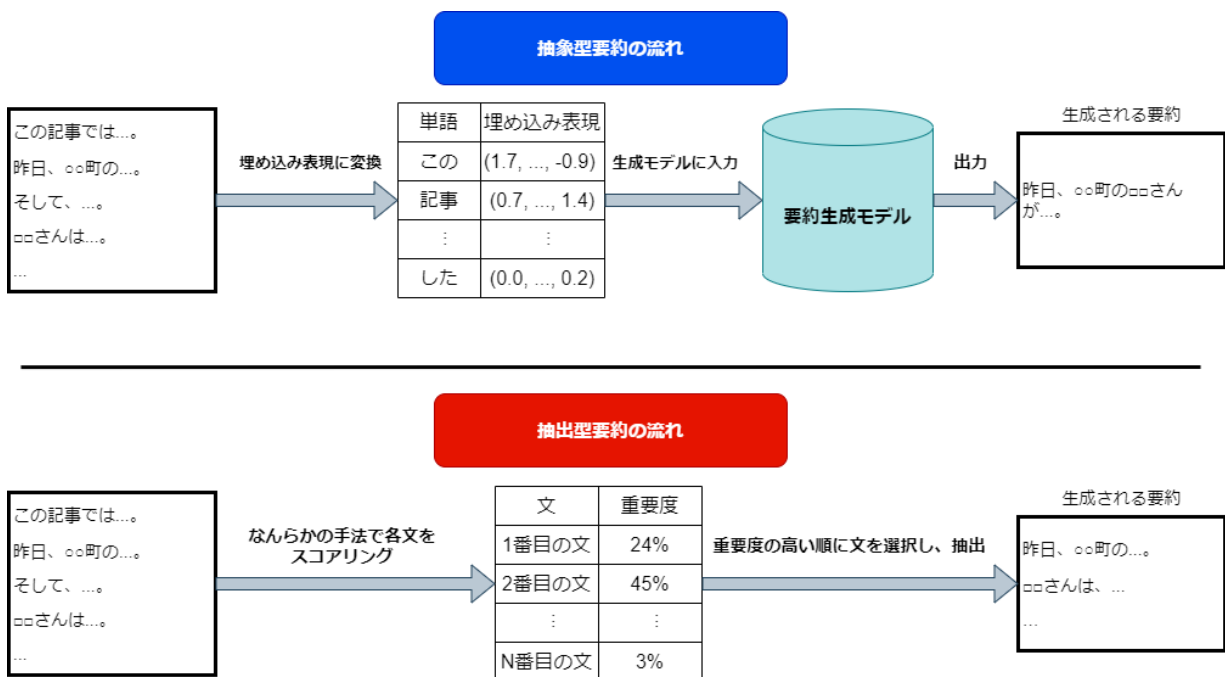


図 2.1 抽出型要約と抽象型要約の違い。抽出型要約では何らかの手法を用いて各文の重要度を決定し、その重要度の結果から文を選択し、それらの文を要約として生成する。対して、抽象型要約では埋め込み表現などの特徴量表現から、生成モデルを用いて要約を生成する。

下に示す。

抽出型要約

長所 本文から抽出して要約を行うため、文法的な違和感が少ない。

短所 抽出した文をそのまま並べて要約とするため、前後の文脈関係が崩れやすい。

抽象型要約

長所 ゼロから生成するため、要約の長さや内容の方向性を調整できる。

短所 生成するモデルによっては文法的に問題がある文章が生成されてしまう。

抽出型要約は古くから研究されており、どのようにして選択する文を決定するか、そしてどのように文を選択していくか、という点において違いがある。選択する文の決定については、文の持つ特徴を用いる手法 [1] や、数理的な手法を用いる手法 [8]、ニューラルネットワークを用いて行う手法 [9] がある。また、文の選択においては単純にスコアリングをして、そのスコアから高い順に選択する手法 [10] や、線形計画法に基づいて選択する手法 [11] も存在する。

対して、抽象型要約は再帰型ニューラルネットワークが登場するまではあまり研究はされていなかった。しかし、その登場以降は様々なモデルが登場し、そして、Transformer、及びそれを用いた事前学習モデル [2] 登場以降は抽象型要約手法による研究が多くなっていった。

本章では 2.3 節にて抽出型要約手法について、2.4 節にて抽象型要約手法についてそれぞれの代表的な手法について解説する。そして、5.3 節にて長文の文書を対象とした自動要約研究について解説する。

2.3 抽出型要約

2.3.1 抽出型要約手法の変遷

抽出型要約の手法の変遷は主に以下のように分けられる。

特徴量を用いる手法

単語の出現頻度や文章の位置などコーパスより得られる特徴を数値化し、それを文選択に使用する手法

数理的な手法

特異値分解や固有ベクトル抽出などを用いてコーパスや言語的特徴に依存しない手法

BERT を用いる手法

大型事前学習モデルである BERT[12] を用いた手法

一番古い手法は特徴量による手法であり、その次に数理的な手法が出現し、そして BERT の台頭によって、それを用いる手法も出現するようになった。

2.3.2 特徴量を用いる手法

特徴量を用いて抽出型要約を行う場合、重要な文の特徴を複数設定し、それをどれだけ含んでいるかをスコアとして用いることが多かった。例えば、以下のような特徴が設定された。

- 文章の長さ
- 単語の出現頻度
- タイトルに含まれる単語と同じ単語をどれだけ含むか
- 文章の位置
- 大文字をどれだけ使用しているか

これらのような、文書から獲得できる特徴をどのように数値化し、どのようにスコアリングに用いるかは手法ごとに大きく異なる。古典的な Lihu のアルゴリズムでは単語の出現頻度に基づいて文のスコアリングを行う。例えば、次のような文があるとする。

例 たかし君はお財布を握りしめて、りんごを買うために出かけた。

Lihu のアルゴリズム [13] では事前に重要な単語をヒューリスティックに決定し、その単語の出現頻度に基づいてスコアリングする。上記の例文において重要な単語を「たかし君」、「お財布」、「りんご」の3つとしたとき、その3つの単語を含む部分を抽出し、その部分の単語数で重要な単語の数の累乗を割って、スコアとする。つまり、例文の場合、抽出される範囲は「たかし君はお財布を握りしめて、りんご」となり、スコアは次式のようなになる。

$$\frac{3^2}{7} \sim 1.2 \quad (3.1)$$

また、発展的な手法として単純ベイズ分類器を用いて行う手法 [14] も存在する。要約対象の文書内のそれぞれの文 s に対して、特徴 F が k 個与えられたとき、その文が要約 S にふくまれる確率 $P(s \in S | F_1, F_2, \dots, F_k)$ というのはベイズの定理を用いて次式で表せる。なお各特徴は独立であるとする。

$$P(s \in S | F_1, F_2, \dots, F_k) = \frac{\prod_{j=1}^k P(F_j | s \in S) P(s \in S)}{\prod_{j=1}^k P(F_j)} \quad (3.2)$$

このとき、分母は定数、 $P(s \in S)$ 、及び $P(F_j | s \in S)$ は学習データより計算することが可能である。なので、最終的には以下の単純ベイズ分類器を得ることができる。

$$\arg \max_s P(s \in S | F_1, F_2, \dots, F_k) = \arg \max_s P(s \in S) \prod_{j=1}^k P(F_j | s \in S) \quad (3.3)$$

なお、多くの場合では $\prod_{j=1}^k P(F_j | s \in S)$ の計算をした結果、計算機においてアンダーフローを

起こすため、対数化して計算を行う。つまり、

$$\arg \max_s \log P(s \in S | F_1, F_2, \dots, F_k) = \arg \max_s \left(\log P(s \in S) + \sum_{j=1}^k \log P(F_j | s \in S) \right) \quad (3.4)$$

上記の式より得られた分類器を用いて、要約を行いたい文書の各文にスコアリングし、そのスコアの高い順に文章を選択していき、要約を生成していく。

特徴量によるヒューリスティックな手法であると同時にシンプルな手法であるが、言語的特徴や要約の対象となる文書の特徴に依存したものが多いため、汎用性には欠けている。

2.3.3 数理的な手法

Latent Semantic Analysis による抽出型要約

数理的な手法の代表的なものの1つとして、Latent Semantic Analysis[15](以下 LSA) による抽出型要約手法がある。LSA はその名の通り、意味解析における手法の1つで、特異値分解を用いて単語間の類似度や、文書間の意味的類似度を計算することができる。また、分解後の特異値行列をトピックとして扱うことで、単語の分類や文書の分類も行うことができる。

LSA による抽出型要約では LSA を文書ではなく、文単位で行い、各トピックにおいて最もそのトピックを表すであろう文を要約として抽出を行う。次から LSA による抽象型要約の手順を解説する。解説における例文として次の3つの文を使用する。

- s_1 たかし君はりんごを買いに行った。
- s_2 りんごを3つ買った。
- s_3 たかし君は八百屋に行った。

まず、上記3つの文に対して、行に単語、列に文を対応させた単語-文行列 D を生成する。なお、行列の成分 D_{ij} は単語の出現回数である。上記の例文を用いた場合、表 3.1 のような行列が生成される。行列を生成したのちは、この行列に対して特異値分解を行う。行列に対する特異値分解は、次式として表される。

$$D = U\Sigma V^T \quad (3.5)$$

この時、 U は左特異値行列、 Σ は特異値行列、 V は右特異値行列と呼ばれる。特異値分解により得られる各行列のサイズは、単語数を m 、文の数を n としたとき、単語-文行列のサイズは $m \times n$ であるので、 U は、 $m \times r$ 、 Σ は、 $r \times r$ 、 V は $n \times r$ となる。このとき、 r は $r = \min(m, n)$ である。また、 U, V はそれぞれ直交行列、 Σ は対角行列である。さらに U, V のそれぞれの列ベクトル \vec{u}, \vec{v} は正規直交基底ベクトルである。各行列の導出は次のように行う。

表 3.1 例文から生成した単語-文書行列. 行列内の成分には列に対応した文章に, 行に対応した単語の出現回数を記録する.

	s_1	s_2	s_3
たかし	1	0	1
君	1	0	1
は	1	0	1
りんご	1	1	0
を	1	1	0
買う	1	1	0
3つ	0	1	0
に	0	0	1
八百屋	0	0	1
行く	0	0	1

式 3.5 より, U, V の列ベクトル \vec{u}_i, \vec{v}_i , 及び Σ の対角成分 σ_i に対して, 次式が成り立つ.

$$\begin{aligned} D\vec{v}_i &= \sigma_i\vec{u}_i \\ D^\top\vec{u}_i &= \sigma_i\vec{v}_i \end{aligned} \quad (3.6)$$

式 3.6 より, $D^\top D$ について, 次の計算が成り立つ.

$$\begin{aligned} D^\top D\vec{v}_i &= D^\top\sigma_i\vec{u}_i \\ &= \sigma_i(D^\top\vec{u}_i) \\ &= \sigma_i^2(\vec{v}_i) \\ &= \sigma_i^2(E\vec{v}_i) \end{aligned} \quad (3.7)$$

このとき, E は単位行列を表す. さらに, 式 3.7 を変形すると, 次の方程式が得られる.

$$(D^\top D - \sigma_i^2 E)\vec{v}_i = 0 \quad (3.8)$$

このとき, \vec{v}_i は正規直交基底ベクトルであるので, $\vec{v}_i \neq 0$ である. よって, 次式を解くことになる.

$$(D^\top D - \sigma_i^2 E) = 0 \quad (3.9)$$

このとき, $D^\top D = X$, $\sigma_i^2 = \lambda_i$ としたとき, 式 3.9 は行列 $X = D^\top D$ の固有値 $\lambda_i = \sigma_i^2$ を求めることと同値である. 固有値 λ_i を計算することで, 対応する右特異値行列の列ベクトル \vec{v}_i も大きさ 1 の固有値ベクトルとして計算可能である. また, 左特異値行列 U の列ベクトル \vec{u}_i は式 3.6 より, 次式から計算可能である.

$$\vec{u}_i = \frac{1}{\sigma_i} D\vec{v}_i \quad (3.10)$$

これを全ての $i = 1, \dots, r$ に対して計算することで U, V, Σ を求めることが可能である。なお、式 3.9 において、 DD^T について解くことでも導出することは可能であり、その際は式 3.9 より得られる固有値ベクトルは左特異値行列の列ベクトル \vec{u}_i の計算となり、右特異値行列の列ベクトル \vec{v}_i は次式より計算する。

$$\vec{v}_i = \frac{1}{\sigma_i} D^T \vec{u}_i \quad (3.11)$$

LSA では、特異値分解より得られる特異値行列は、トピック-トピック行列とも言う。このトピックというのは単語間、文書間、もしくは単語-文書間における潜在的な意味、つまりは話題やジャンルといったものを指す。また、LSA では右特異値行列は文書-トピック行列、左特異値行列は単語-トピック行列とも言う。右特異値行列を解析することで、文書と話題の関連性を調べたり、左特異値行列を用いることで単語の意味的分類を行うことが可能である。単一文書に対する LSA では、 V は行が文、列がトピックに該当するため、 V を用いて抽出型要約を行う。例文の場合、得られる右特異値行列は表 3.2 になる。そして、各列ごとに値の高い文を重複がないように選

表 3.2 表に対して特異値分解を行って得られる右特異値行列。行が文、列がトピックに対応する。

	トピック 1	トピック 2	トピック 3
s_1	-0.732	-0.374	-0.568
s_2	0.641	-0.659	-0.391
s_3	0.228	0.651	-0.723

択する。例えば、2 文まで選択する場合は、 s_2 と s_3 が該当する。なので、最終的な要約は「りんごを買いに行った。たかし君は八百屋に行った。」となる。

この LSA による抽出型要約は、LSA 以外の潜在的意味解析を用いても行うことができる。例えば、確率モデルによる潜在的意味解析である Latent Dirichlet Allocation[16](以下 LDA) を LSA の代わりに適用することもできる。LDA では文書を単語の集まりとして定義し、単語はトピックから生成されるものとしている。具体的には LDA では文書の生成過程を以下の手順であると定義する。

1. 文書内の単語数 N をポアソン分布で決定する。
2. α をパラメータとするディリクレ分布 $Dir(\alpha)$ からトピックについての多項分布のパラメータ θ を決定する。
3. N 個の各単語 w_n に対して、
 - (a) w_n に対する潜在的トピック z_n を多項分布 $Multi(\theta)$ より決定する。
 - (b) 潜在的トピック z_n に関する条件付き確率 $p(w_n|z_n, \beta)$ より単語 w_n を決定する。

このとき、 β は各トピックにおける単語の生成確率を表す行列である。上述の生成過程に基づいて、近似推定などを用いて、各パラメータを学習する。LDA を LSA の代わりに抽出型要約に使

用する際には、文書内の各文それぞれを1つの文書としてみなすことで、ディリクレ分布のパラメータ α を文書のトピック分布、多項分布のパラメータ θ を文のトピック分布としてみなすことができるため、各文のトピック分布 θ を右特異値行列の代わりに使用して抽出型要約を行うことも可能である。

LexRank

ほぼ同時期にグラフ構造による手法として LexRank も提案 [17] された。この手法では、文同士の類似度を元に隣接行列を作成し、隣接行列より遷移確率行列を計算する。この遷移確率行列の固有値ベクトルが各文のスコアとなり、値の大きいものが重要な文として選択される。類似度の計算には、tf-idf 法とコサイン類似度を使用する。tf-idf 法とは単語の出現頻度に基づいた特徴量の1つであり、tf 法と idf 法より得られる値の積で計算される。tf 法は単純に単語の出現頻度を表すが、1文の中での出現頻度を指す。つまり、ある文 s_j における単語 w_i の tf 法による出現頻度 $TF(w_i, s_j)$ は次式より計算される。

$$TF(w_i, s_j) = \frac{m(w_i, s_j)}{\sum_{w' \in s_j} m(w', s_j)} \quad (3.12)$$

$m(\cdot)$ は出現回数を表す。対して、idf 法は文書全体で出現した回数に対して、全文の数で割った値の逆数の対数を表す。つまり、idf 法では要約対象の文書が N 個の文があり、文書に含まれるある単語 w_i に対して、次のように計算する。

$$IDF(w_i) = \log \frac{N}{n_{w_i} + 1} + 1 \quad (3.13)$$

n_{w_i} は単語 w_i が出現した文の数を示す。これを用いて各文ごとに単語の tf-idf 法による特徴量を計算し、文それぞれの特徴量ベクトルを作成する。さきの例文 s_1, s_2, s_3 を用いて行う場合、それぞれの特徴量ベクトルは次の通り。

$$\begin{aligned} s_1 & (0.264, 0.264, 0.264, 0.264, 0.264, 0.264, 0.715, 0.264, 0.0, 0.0, 0.0, 0.0) \\ s_2 & (0.0, 0.0, 0.0, 0.296, 0.296, 0.0, 0.0, 0.296, 0.804, 0.296, 0.0, 0.0) \\ s_3 & (0.225, 0.225, 0.225, 0.0, 0.0, 0.0, 0.225, 0.0, 0.0, 0.0, 0.225, 0.610, 0.610) \end{aligned}$$

そして各文の類似度を計算すると表 3.3 のとおりとなる。

表 3.3 tf-idf 法より得られた各文同士の類似度行列。

	s1	s2	s3
s1	1.0	0.235	0.238
s2	0.235	1.0	0.06
s3	0.238	0.06	1.0

表 3.3 の類似度から表の隣接行列を計算し、そこから遷移確率行列 (表) を生成する.

表 3.4 表 3.3 の類似度行列より得られる隣接行列. 閾値を 0.24 として、閾値より高い値を隣接しているものとした.

	s1	s2	s3
s1	1	0	1
s2	0	1	0
s3	1	0	1

表 3.5 隣接行列から計算した遷移確率行列.

	s1	s2	s3
s1	0.5	0.0	0.5
s2	0.0	1.0	0.0
s3	0.5	0.0	0.5

最後に遷移確率行列から最大固有値に対応する、大きさ 1 の固有値ベクトルを計算する. 固有値ベクトルの計算を行う際、場合によっては計算するのが困難な場合がある. そのため、色々な計算方法があるが、LexRank ではべき乗法による計算を行う. べき乗法は正方行列の固有値の中で最大のものを求める手法であり、同時にそれに対応する固有値ベクトルを求めることができる. べき乗法では次の前提条件がある.

- 正方行列 A の固有値 λ_i ($i = 1, \dots, n$) は重複しない.
- 固有値には次の関係が成り立つ: $|\lambda_1| > \dots > |\lambda_n|$
- 各固有値ベクトルは一次独立.

このとき、任意のベクトル b を A の固有値ベクトルを用いて表す場合、固有値ベクトル x_i ($i = 1, \dots, n$) の線形結合で表現することが可能である. つまり、次のように表現可能である.

$$b = c_1 x_1 + \dots + c_n x_n \quad (3.14)$$

べき乗法ではこのベクトルを用いて、次式のような繰り返しを行う.

$$\begin{aligned} b^{(1)} &= Ab \\ b^{(2)} &= Ab^{(1)} = A^2 b \\ &\vdots \\ b^{(k)} &= Ab^{(k-1)} = A^k b \end{aligned} \quad (3.15)$$

式 3.15 から $b^{(k)}$ は, $Ax_i = \lambda_i x_i$ より, 次式として展開できる.

$$\begin{aligned} b^{(k)} &= c_1 \lambda_1^k x_1 + \cdots + c_n \lambda_n^k x_n \\ &= \lambda_1^k \left\{ c_1 x_1 + \cdots + c_n \left(\frac{\lambda_n^k}{\lambda_1^k} \right) x_n \right\} \end{aligned} \quad (3.16)$$

このとき, 式 3.16 において, $k \rightarrow \infty$ とすると, 前提条件より, $\frac{\lambda_i^k}{\lambda_1^k} \rightarrow 0$ ($i = 2, \dots, n$) となるため, 次のようになる.

$$b^{(k)} \rightarrow c_1 \lambda_1^k x_1 \quad (3.17)$$

このとき, 固有値ベクトルは定数倍しても固有値ベクトルであるので,

$$b^{(k)} \rightarrow x_1 \quad (3.18)$$

となり, 最終的には最大固有値の固有値ベクトル x_1 に収束していく. このことから, べき乗法においては次式のように固有値ベクトルを更新する.

$$x_1^{(k)} := Ax_1^{(k-1)} \quad (3.19)$$

LexRank においては, 最大固有値ベクトルの初期値は $x_1 = \frac{1}{N} \mathbf{1}$ としている. また, 実際には有限の回数で計算を行う必要があるため, 収束判定を用意する必要がある. LexRank では, 収束判定の閾値を ϵ として, 次を満たすまで更新を続ける.

$$\|x^{(k)} - x^{(k-1)}\| < \epsilon \quad (3.20)$$

つまり, LexRank において, 最大固有値ベクトルをアルゴリズム 2 で求める.

Algorithm 2: LexRank におけるべき乗法による最大固有値ベクトルの獲得

$A \leftarrow$ 遷移確率行列;

$x_0 \leftarrow \frac{1}{N} \mathbf{1}$;

$t \leftarrow 0$;

repeat

$t = t + 1$;

$x_t = A^\top x_{t-1}$;

$\delta = \|x_t - x_{t-1}\|$;

until $\delta < \epsilon$;

return x_t

表 3.5 に対して, べき乗法を適応して得られる最大固有値に対応する固有値ベクトルは (0.33, 0.33, 0.33) である. 例文の場合, どれも等しい値であるが, 2 文から構成される場合には先

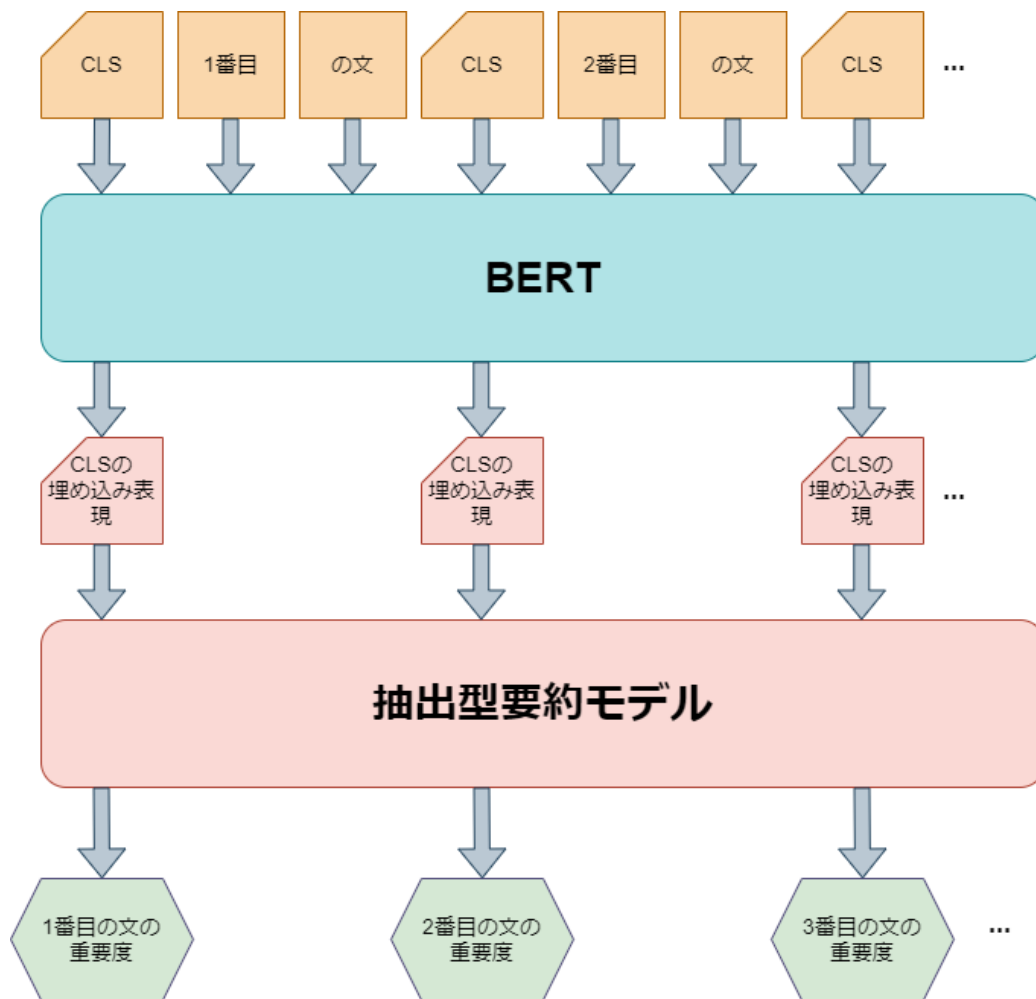


図 3.1 BERTSum における要約生成の過程. BERT を通して出力された CLS トークンの埋め込み表現を BERT とは別の抽出型要約モデルに入力して, 各文章の重要度を獲得する. この重要度が高いものから順に文を選択していく.

頭から選択するため, s_1, s_2 が選択される. よって, 最終的な要約文は「たかし君はりんごを買いに行った. りんごを3つ買った。」となる.

2.3.4 BERT を用いる手法

BERT とは Transformer[18] と呼ばれるモデルを用いた大型の事前学習モデルである. より詳細に言えば, Transformer 内の入力データの特徴量変換を担当する Encoder のみを用いたモデルである. BERT と Transformer についての詳細は抽象型要約の節にて解説を行う.

BERT を用いた抽出型要約の代表的な手法として BERTSum がある [19]. BERTSum では図 3.1 のような形で BERT に要約を行いたい文書を入力する. BERTSum では図 3.1 に示してい

る CLS トークンに対応する出力 T_i を抽出型要約を行うモデルに入力する。要約を行う側の出力は各文が抽出対象となる確率を表しており、この確率が高い順に選択していく。

2.4 抽象型要約

2.4.1 抽象型要約の台頭

抽象型要約は抽出型要約と比較して、ニューラルネットワークが登場するまではあまり研究が進んでいない状態であったが、再帰型ニューラルネットワーク (以下 RNN) が出現してからは研究が進み、この RNN を 2 つ繋げた Encoder-Decoder モデルが出現した。さらに Transformer の出現によって、より文法的に違和感のない文章の生成が可能となった。これにより、抽出型要約ではなく、現在の自動要約生成の研究の主流は抽象型要約へ変化していった。

文章生成モデルにおける基本となる RNN[20] とそれを用いた Encoder-Decoder モデル [21]、さらに要約における有名な Encoder-Decoder モデルである Pointer-generator Network[22]、最後に Transformer とそれを用いたモデルについて解説する。

2.4.2 RNN

RNN は系列データを取り扱うニューラルネットワークであり、その名の通り、系列データの時間方向の変化に合わせて、隠れ層の状態も前の時間の状態を用いて更新する。RNN の構造を図 4.1 に示す。図の通り、RNN では時間 t の隠れ層の状態 h^t を次式で更新していく。

$$h^t = W_{xh}x^t + W_{hh}h^{t-1} \quad (4.1)$$

x^t は時間 t における入力、 W_{xh}, W_{hh} はそれぞれ重みである。これによって、前の時間の情報を次の時間に組み込みながら、出力を得ることができる。また、RNN の発展モデルとしてより長い系列データの情報の記憶が可能な LSTM や、GRU というモデルがある。

RNN の問題点

RNN における問題点として、長期の系列データによって学習を行った場合、勾配降下法による誤差逆伝搬にて勾配が消失、もしくは爆発してしまうという問題がある。この問題は順伝搬ネットワークにおいても同様に発生する。例えば順伝搬ネットワーク内における、出力層ニューロン u から隠れ層のニューロン v に対する重み W_{uv} における、勾配降下法による誤差逆伝搬は誤差関数を E としたとき、

$$\frac{\delta E}{\delta W_{uv}} = \frac{\delta E}{\delta y_u} \frac{\delta y_u}{\delta s_u} \frac{\delta s_u}{\delta W_{uv}} \quad (4.2)$$

となる。このとき、 y_u がユニット u の状態 s_u を活性化させた状態、つまりは活性化関数を f とし、 $y_u = f(s_u)$ とする。このとき、活性化関数をシグモイド関数 $\sigma(s_u)$ としたとき、その s_u に

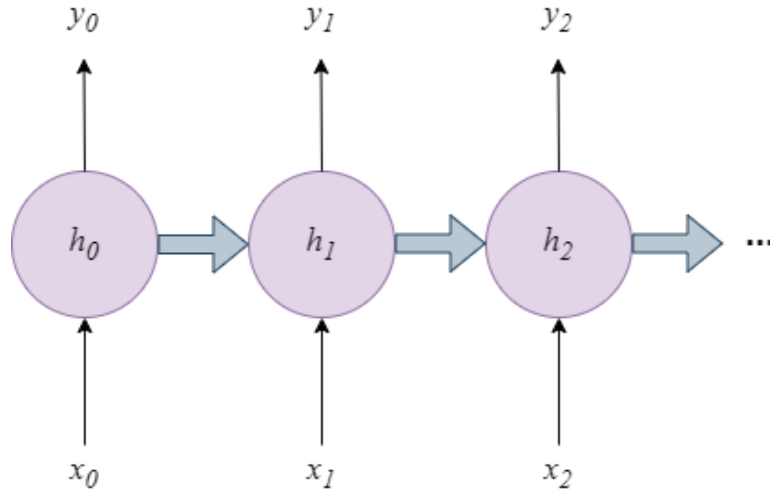


図 4.1 RNN モデルにおける内部状態の図. 時間ごとに隠れ層である h の状態を, ひとつ前の h の状態と, 入力である x を用いて更新する. これにより, 系列データの入力と学習が可能になり, 時間ごとの出力を得ることもできる.

よる微分 $\sigma'(s_u)$ は次式となる.

$$\sigma'(s_u) = (1 - \sigma(s_u))\sigma(s_u) \quad (4.3)$$

シグモイド関数は必ず 0 以上 1 未満であるため, シグモイド関数の微分も同様に 0 以上 1 以下となる. このとき, 例として, 入力層のニューロン i から隠れ層ニューロン v への重みの更新式 W_{iv} は,

$$\frac{\delta E}{\delta W_{iv}} = \frac{\delta E}{\delta y_u} \frac{\delta y_u}{\delta s_u} \frac{\delta s_u}{\delta y_v} \frac{\delta y_v}{s_i} \frac{s_i}{w_{iv}} \quad (4.4)$$

と拡張できる. このとき, 入力層から隠れ層への活性化関数もシグモイド関数であるとき, このとき, $\frac{\delta E}{\delta W_{uv}} > \frac{\delta E}{\delta W_{iv}}$ が成り立つ. つまり, 重みの更新に使われる勾配の値が減少していることがわかる. より大きな層のニューラルネットワークでは勾配の値がより 0 に近づくことになり, 勾配が消失してしまう. これが勾配消失問題と呼ばれ, 逆に活性化関数の微分が必ず 1 よりも大きい場合に発生するのが勾配爆発問題である. 通常の順伝搬ネットワークの場合, 出力以外の活性化関数を ReLU 関数に置き換えることで対応できる. ReLU 関数 $Relu$ は次の式で表される.

$$Relu(x) = \max(0, x) \quad (4.5)$$

この ReLU 関数は非常に単純な関数で, 0 未満の場合は 0 を出力し, 0 より大きければそのまま出力するという関数であるため, ReLU 関数の微分 $ReLU'(x)$ は次式の通りとなる.

$$ReLU'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{(if) } x \leq 0 \end{cases} \quad (4.6)$$

つまり、ReLU 関数を活性化関数として使用する場合、誤差逆伝搬において勾配消失及び、勾配爆発が発生しない。

対して、RNN の場合、順伝搬ネットワークのように出力層から入力層までの誤差逆伝搬だけではなく、時間方向の誤差逆伝搬を考慮する必要がある。RNN において、ある隠れ層 h において、時間 t から t' 間についての勾配は次式で表される。

$$\prod_{\tau=t+1}^{t'} f'(z_h(t' - \tau + t)W_{h_\tau, h_{\tau+1}}) \quad (4.7)$$

なお f は活性化関数、 z_h は隠れ層 h における加重和、 $W_{h_\tau, h_{\tau+1}}$ は RNN における重み更新に使用する重みである。この時、式 4.7 が 1 より小さければ勾配消失、1 より大きければ勾配爆発が発生する。

これを解決する方法として、提案されたのが定誤差カルーセル (以下 CEC) というものである。式 4.7 において、勾配爆発、もしくは消失が発生しないようにするには、

$$\prod_{\tau=t+1}^{t'} f'(z_h(t' - \tau + t)W_{h_\tau, h_{\tau+1}}) = 1 \quad (4.8)$$

として、勾配を固定すればよい。これが CEC である。勾配を固定するという事は、誤差をそのまま伝播するという事でもあるため、誤差を記憶することができる。実際には、時間方向に出力する際に活性化関数として恒等関数を使用し、加重和に使用する重みを 1.0 とすることで実現する。

RNN にはさらにもう一つの問題点がある。それは、時系列ごとの入力に関する情報と、出力に関する情報をそれぞれ 1 つの重みで対応する必要があるということである。具体的には、入力においてどの時間で重要な情報が来て、重要でない情報が来るのか、ということ学習を 1 つの重みで学習する必要がある。これは出力においても同様である。そのため、入力の情報を用いてどの入力を通すべきか、どの出力を行うべきかを判定する入力ゲートと出力ゲートを追加することで、対応する。時間 t における入力ゲートの出力 In_t は、

$$In_t = f(W_{in}^x x_t + W_{in}^h h_{t-1} + b_{in}) \quad (4.9)$$

として表される。 f は活性化関数、 W_{in}^x, W_{in}^h はそれぞれ入力に対する重み、 $t-1$ 時間前の隠れ層の出力、 b_{in} は入力ゲートにおけるバイアスである。出力ゲートにおいても、同様の計算を行う。加えて、CEC における誤差情報を入力の情報に基づいてリセットしたい場合もある。それを実現するため、CEC の状態更新において、その情報を伝播するための忘却ゲートも追加する。時間 t における忘却ゲートも入力ゲートと同様の計算を行う。

LSTM

LSTM は, Long-Short Term Memory の略で, 先述した以下の 4 つの要素を実装した RNN である [23].

- CEC
- 入力ゲート
- 出力ゲート
- 忘却ゲート

LSTM では, RNN とは異なり, 時間 t において, 出力ゲートからの出力 h_t を次の時間に出し, 加えて CEC による長期記憶を表す出力 c_t を時間方向に出し, 時間 t にて, 入力として x_t が与えられたとき, それぞれの出力は次式より計算される.

$$\begin{aligned}h_t &= Out_t \otimes \sigma(c_t) \\Out_t &= \sigma(W_{out}^x x_t + W_{out}^h h_{t-1} + b_{out}) \\c_t &= Fog_t \otimes c_{t-1} + In_t \otimes \tanh(W_c^x x_t + W_c^h h_{t-1} + b_c) \\In_t &= \sigma(W_{in}^x x_t + W_{in}^h h_{t-1} + b_{in}) \\Fog_t &= \sigma(W_{fog}^x x_t + W_{fog}^h h_{t-1} + b_{fog})\end{aligned}\tag{4.10}$$

In_t, Out_t, Fog_t はそれぞれ, 入力ゲート, 出力ゲート, 忘却ゲートを示す. LSTM モデルの概略図を図 4.2 に示す.

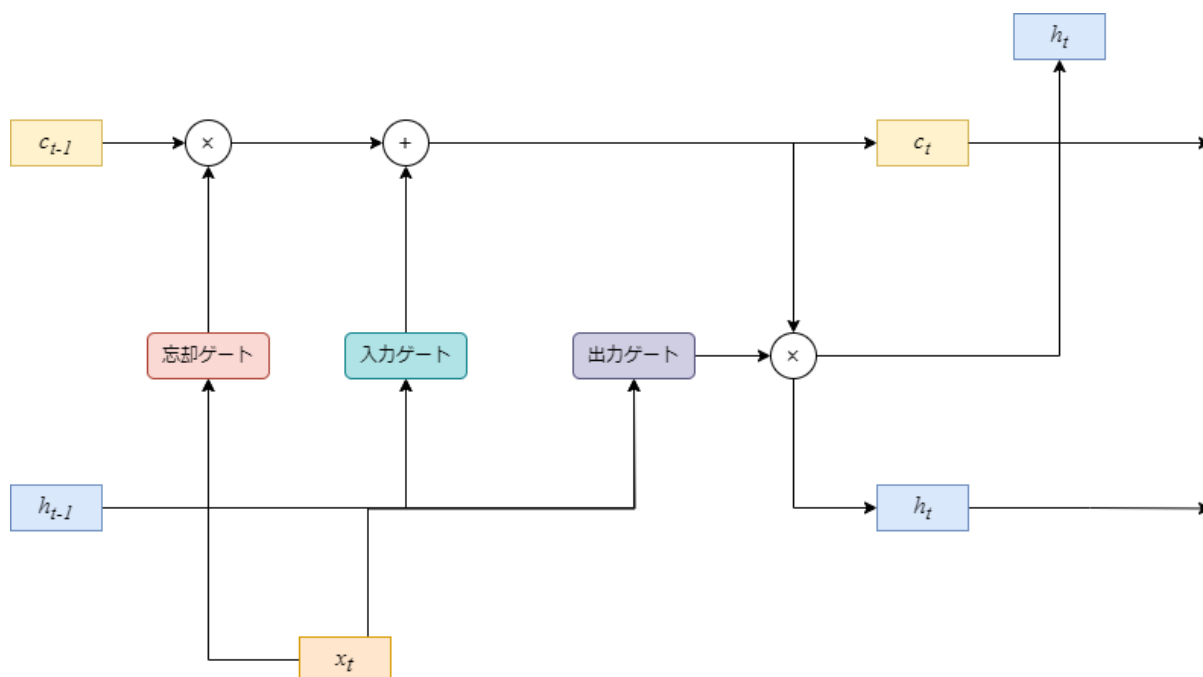


図 4.2 LSTM モデルの構造図. 前の時間における CEC による長期記憶 c_t とモデルの出力 h_t が伝播していく. 特に長期記憶については勾配は常に忘却ゲートのみとなるため, 忘却ゲートの出力が 1 に近くなるようにバイアスで調整すれば勾配爆発や勾配消失が起きなくなる.

伝播する誤差の情報や情報の伝達を時間ごとに調整することが可能となっている. さらにこの発展として, 誤差の情報も入力, 出力, 忘却ゲートに使用する LSTM もある.

また別の発展モデルとして Gated Recurrent Unit (GRU) と呼ばれるモデルもある [24]. LSTM では各ゲートの計算と, 図からわかる通り, CEC の更新とは別に各ゲート用の重みの計算を行う必要があり, 計算コストが高いという問題がある. GRU では, ゲートが更新ゲートとリセットゲートの 2 つに減少し, 長期記憶とモデルの出力を統一することでコストを減らしている.

この LSTM モデル及び GRU モデルは, Transformer の登場までにおける主要な系列モデルであり, 特に文章生成において使用されたモデルである. このような再帰型ニューラルネットワークを用いた文章生成モデルについて以下解説していく.

2.4.3 Encoder-Decoder モデル

Encoder-Decoder モデルは, 系列データを入力として受け取り, 更新した RNN の隠れ層の出力を元に別のデータを生成するモデルである. 自動要約では主に系列データから別の系列データを生成する, sequence-to-sequence [21] (以下 seq2seq) の構造が一般的である. seq2seq の構造を可視化すると図 4.3 の通りである. 図の左側が入力となる系列データを受け取る Encoder であり, 右側が Encoder からの隠れ層の状態を受け取り別の系列データを生成する Decoder に該当

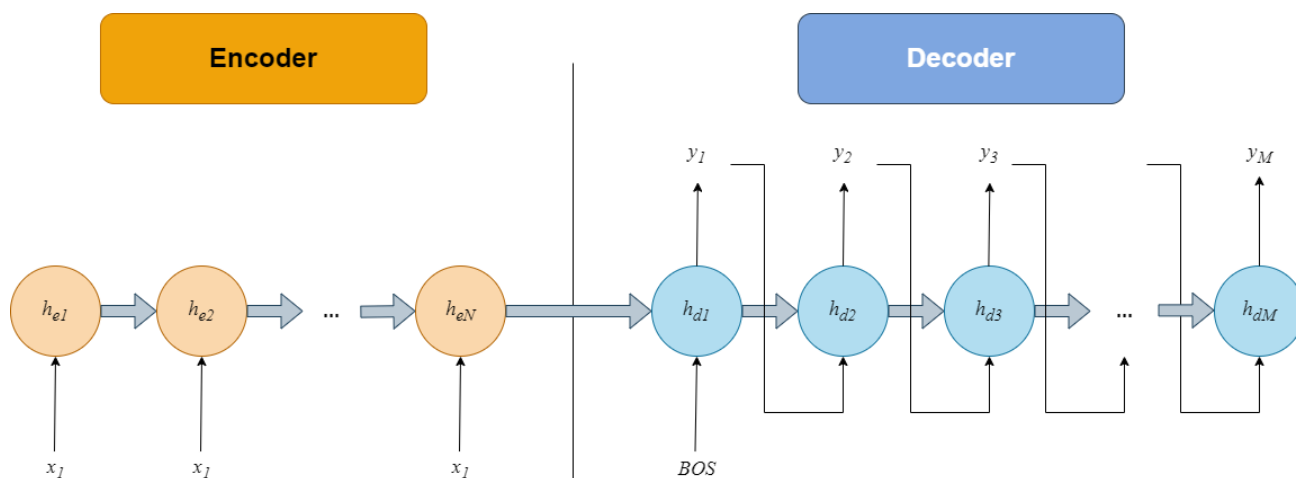


図 4.3 Encoder-Decoder モデルの一つである sequence-to-sequence モデルの概略図。Encoder 側では入力する系列データを生成し、最終入力時の隠れ層の状態を出力する。そして、Decoder 側では、開始記号と Encoder が出力してきた隠れ層の状態を受け取り、系列データの生成を行う。

する。Encoder 側では RNN を用いて隠れ層の更新をして、隠れ層の状態を、入力データの情報として出力するが、RNN の出力層は Decoder 側に渡さない。Decoder 側は Encoder 側の隠れ層の状態と、開始記号を入力し、系列データの生成を行う。系列データの生成時、Decoder 側では出力層からデータを生成するが、このデータを次の時間の入力に使用して、次の系列データの生成に使用するという形で繰り返し生成していく。生成を止めるタイミングはモデルにより様々ではあるが、基本的には終端記号が生成された際に、生成を停止する。

この Encoder-Decoder モデルは RNN だけでなく、Transformer でも同様の構造が採用されており、現在の生成モデルの一般的な構造として用いられている。

sequence-to-sequence モデルにおける文章生成

seq2seq を用いた文書生成モデルの紹介をする前に、文書生成モデルにおける生成過程と学習について解説する。seq2seq モデルを用いた文章生成は図 4.3 に沿って、Encoder 側に生成のために必要な情報を持つ文書を 1 単語ずつ入力する。Encoder 側への入力には、例えば翻訳であれば翻訳元の文、要約であれば要約したい文書が該当する。

Decoder 側における各時間の出力は単語となる。各時間の出力を結合し、一つの文、もしくは文書となるようにモデルの構築、及び学習を行う必要がある。そのため、seq2seq を用いた文章生成では、Decoder における最終出力では、単語分布と呼ばれる、ある時間における単語の生成される確率分布を出力する。この確率分布を生成するために、seq2seq による文章生成モデルでは最終出力における活性化関数として、ソフトマックス関数と呼ばれる関数を使用する。このソフト

マックス関数 $\text{softmax}(\cdot)$ では、入力として長さ N のベクトル X の要素 x_i を受け取り、次式の計算によってベクトルの要素 x_i を変換する。

$$\text{softmax}(x_i) = \frac{x_i}{\sum_{j=1}^N x_j} \quad (4.11)$$

この式からわかる通り、ソフトマックス関数では入力したベクトルに対して、各要素の総和が 1 になるように変換する。そのため、活性化関数としてソフトマックス関数を用いた場合、その出力は確率分布として考えることができる。そのため、出力層の状態を、出力層の前の隠れ層からの出力ベクトルの大きさを単語の語彙数に線形変換するようにし、ソフトマックス関数を用いて変換することで単語分布を得ることが出来る。

文書生成モデルにおける学習では、損失関数に負の対数尤度を用いる。負の対数尤度は、確率分布のパラメータ推定手法の一つ、最尤推定における尤度として用いられる。尤度とは、ある確率分布のパラメータが決まっている場合、そのパラメータの下であるデータを得たとき、そのデータを得る確率のことを指す。実際には次のように計算する。ある確率分布に従う M 個のデータ d_i ($i = 1, \dots, i, \dots, M$) があるとき、この確率分布の密度関数を $f(\cdot)$ としたとき、尤度 L は、

$$L = \prod_{i=1}^M f(d_i) \quad (4.12)$$

より得ることができる。この尤度に対して、対数化を行い、負の値になるようにマイナスを掛けたものが負の対数尤度である。つまり、次式の通りとなる。

$$L = - \sum_{i=1}^M f(d_i) \quad (4.13)$$

文章生成モデルでは、ある時間における正解となる単語 w_t に対して、負の対数尤度を損失関数として適用する。つまり、損失関数 $loss$ は次式を用いる。

$$loss = - \sum_{t=1}^T f(w_t) \quad (4.14)$$

このときの確率密度関数はソフトマックス関数である。

尤度によって、正解データの文書がモデルより生成される確率を得ることができる。事実、文書が単語が時系列順に並んだものであるとした場合、ある時間 t における正解となる単語 w_t が生成される確率を $P(w_t)$ としたとき、正解の文章が生成される確率 $P(S)$ は

$$P(S) = \prod_t P(w_t) \quad (4.15)$$

と表される。このとき、生成確率はソフトマックス関数より得るため、これを対数化し、マイナスを掛けることで損失関数 $loss$ を得ることができる。このとき、各単語の生成される確率が 1 に近

ければ損失関数は 0 に近づいていく。そのため、単語の生成される確率に対する負の対数尤度を用いることで、文章生成モデルの学習を行うことが可能となる。この損失関数のイメージを図 4.4 に示す。



図 4.4 文章生成における負の対数尤度の誤差のイメージ。赤い太字の単語が目的の単語となる。「りんご」が生成されてほしい単語の場合、理想としては右側の単語分布が出力されてほしいが、実際に Decoder から出力される場合、左側のような単語分布が生成される。そのため、理想の単語分布に近づけるようにその確率の差を埋めていくように学習する。

つまり、各時間において目的の単語が得られるように Decoder より出力される確率分布を調整していく。

2.4.4 Pointer-generator Network

Transformer 登場以前にて、seq2seq による Encoder-Decoder モデルを用いた自動要約モデルの代表的なものとして、Pointer-generator Network[22] が存在する。このモデルでは通常の RNN による Encoder-Decoder モデルとは異なり、Attention[25] と呼ばれる機構と Coverage と呼ばれる機構を持つ。Attention とは入力 of 文書内の単語と出力される単語の関連性について学習する機構であり、Coverage とは生成される要約文について、単語の過剰な重複を防ぐための機構である。さらに、Pointer-generator Network にはモデルの持つ語彙に含まれない単語に対応するための機構も含まれている。ここから、各機構について解説する。

Attention

Attention では Decoder の各時間 t ごとに Attention 分布 a_i^t を計算する. この Attention 分布 a_i^t は, Encoder の入力 i における隠れ層の状態 e_i と, Decoder 側の時間 t における Decoder の状態 s_t を用いて, 次のように計算する.

$$\text{score}(e_i, s_t) = v^\top \tanh(W_{e_i}e_i + W_{s_t}s_t + b) \quad (4.16)$$

$$a_i^t = \text{softmax}(\text{score}(e_i, s_t)) \quad (4.17)$$

なお, $\text{softmax}(\cdot)$ はソフトマックス関数であり, 長さ N のベクトル X の各要素 (x_1, \dots, x_N) に対して, 以下の式で計算する.

$$\text{softmax}(x_i) = \frac{x_i}{\sum_{j=1}^N x_j} \quad \text{for } i = 1, 2, \dots, N \quad (4.18)$$

また, v, W_{e_i}, W_{s_t} はそれぞれ重みである. 式 4.18 からわかる通り, 最終的な a_i^t は各要素の総和が 1 になるような確率分布と取れる. そして a_i^t と e_i を用いて, Context ベクトル c^t と呼ばれるベクトルを計算する.

$$c^t = \sum_{i=1}^K a_i^t e_i \quad (4.19)$$

式 4.19 では Attention 分布と Encoder の各隠れ層の状態の内積を取ることで, 時間 t における必要な Encoder の状態を Decoder 側の出力に使用することができる.

Coverage

Coverage は Attention 分布を用いて, 同じ Attention 分布が生成されないように学習をさせることで, 単語の重複を回避する機構である. Coverage では, Decoder 側の時間 t での出力において, Coverage ベクトル c_i^{*t} を次式より計算する.

$$c_i^{*t} = \sum_{t'=0}^{t-1} a_i^{t'} \quad (4.20)$$

そして得られた c_i^{*t} を用いて式 4.16 を次のように書き換える.

$$\text{score}(e_i, s_t) = v^\top \tanh(W_{e_i}e_i + W_{s_t}s_t + W_{c_i^{*t}}c_i^{*t} + b) \quad (4.21)$$

これによって, 時間 t までの Attention 分布の情報を加味することができる. さらに, 学習時の損失関数として時間 t ごとに covloss_t を次式より計算する.

$$\text{covloss}_t = \sum_i \min(a_i^t, c_i^{*t}) \quad (4.22)$$

これにより, 過去と同じ Encoder の隠れ状態に対して重みを与えていた場合により損失関数が増加していく.

Pointer-generator Network における単語生成

Pointer-generator Network では単語を Decoder より生成する際に、モデルの語彙に含まれる単語を出力する確率についての分布 P_{vocab} を計算する.

$$P_{vocab} = \text{softmax}(V'(V[s_t, c_t] + b) + b') \quad (4.23)$$

そして、Pointer-generator Network では単語を生成する確率 p_{gen} を次式から計算する.

$$p_{gen} = \sigma(w_{ct}c^t + w_{st}s_t + w_{yt}y_t + b_{gen}) \quad (4.24)$$

ここで、 $w_{ct}, w_{st}, w_{yt}, b_{gen}$ は学習可能なパラメータであり、 $\sigma(\cdot)$ はシグモイド関数であり、ある値 z に対して次式のように計算される.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (4.25)$$

そして、Decoder における時刻 t にて、ある単語 w が出力される確率 $P(w)$ は次から計算する.

$$P(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (4.26)$$

単語 w が語彙に含まれずに、入力された文書に含まれていた場合、前項は 0 になり、同様に w が語彙に含まれているが、入力された文書に含まれていない場合は後ろの項は 0 となる。これにより、モデルの語彙に含まれていない単語も Decoder からの出力対象として考慮することができる。

最後に図 2.4.4 にモデルの全体図を掲載する。図における、「最終的に出力する単語の分布」が Pointer-generator Network における Decoder の最終出力である。この単語分布にて語彙に含まれていない”たかし”を出力対象として考慮されていることがわかる。

2.4.5 Discourse-aware Attention Model

抽象型要約では、論文の構造をニューラルネットワークに落とし込んだ Discourse-Aware Attention Model[5] が Arman らが提案している。このモデルは先述した Pointer-generator Network と類似しているが、大きく異なる点として Encoder 側の構造と、それに伴う Attention 分布の計算である。

まず、Encoder 側の構造は 2 つの Encoder より構成されており、節の $EncoderRNN_{sec}$ と文書の $EncoderRNN_{doc}$ の 2 つである。これらは共に再帰型ニューラルネットワークである。とある要約の対象の論文が N 個の節からなり、1 つの節あたり単語数が M であるとする。このとき、まず RNN_{sec} より節ごとに節の埋め込み表現を得る。ある j 番目の節の埋め込み表現 $h_j^{(s)}$ 及び、

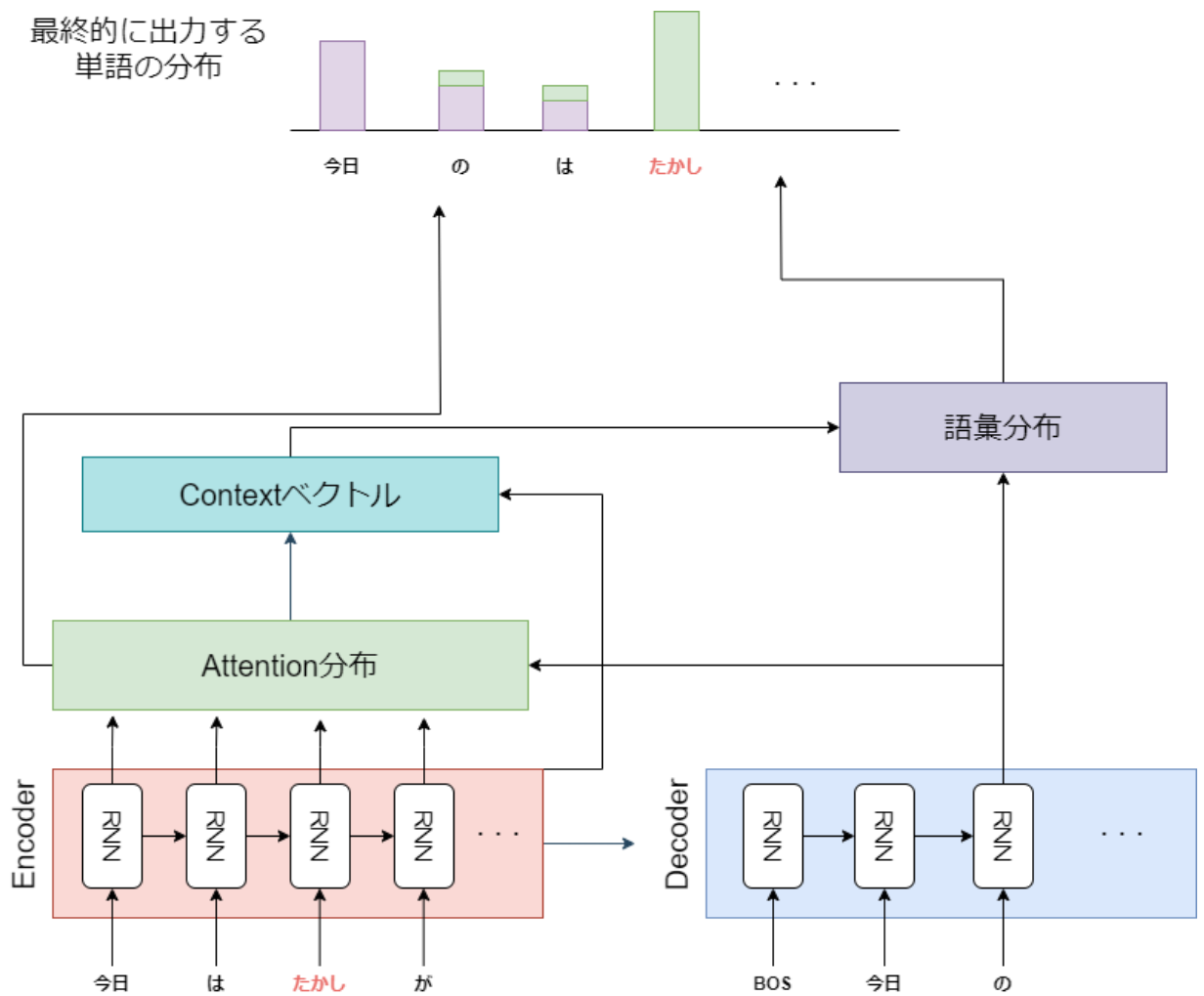


図 4.5 Pointer-generator Network の概略図. 図の中の赤字はモデルの語彙に含まれていない単語を指す. Pointer-generator Network では Attention 分布をさらに最終出力で使用することで, モデルの語彙にない単語を入力側からコピーすることができる. これにより, 固有名詞などのモデルの語彙に含まれている単語を要約の生成に使用することが可能となる.

論文の埋め込み表現 d は次式より獲得する.

$$h_j^{(s)} = RNN_{sec}(x_{(j,1)}, \dots, x_{(j,i)}, \dots, x_{(j,M)}) \quad (4.27)$$

$$d = RNN_{doc}(h_1^{(s)}, \dots, h_j^{(s)}, \dots, h_N^{(s)}) \quad (4.28)$$

ここで $x_{(j,i)}$ は j 番目の節における i 番目の単語における埋め込み表現を指す. これを可視化すると図 4.6 のようになる.

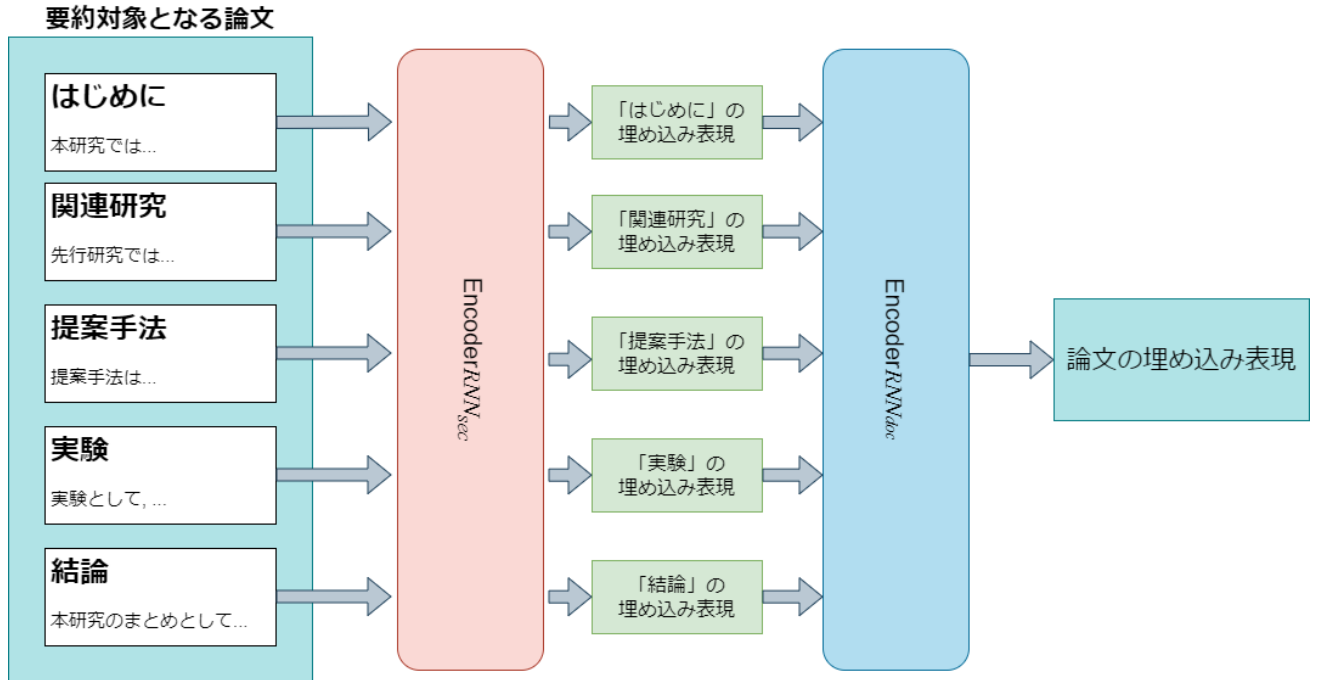


図 4.6 Arman らが提案した Discourse-Aware Attention Model の Encoder 側の構造. 各節を埋め込み表現に変換してから論文全体の埋め込み表現を生成する.

つまり、セクションごとに埋め込み表現を得ることで、論文を対象とする要約における単語数の問題を解決するだけでなく、セクション別の情報を用いた文書情報を得ることもでき、さらにセクションごとの情報も埋め込み表現として獲得できるという利点がある。

次に Attention 分布の計算について説明する。先の Encoder で得られた埋め込み表現を用いて、節ごとの Attention 分布を計算し、さらにそれを用いて単語ごとの Attention 分布を計算する。まず、節の Attention 分布を計算する。ある時間 t における j 番目の節に対する Attention 分布 $\beta_j^{(t)}$ は次式より計算する。

$$\beta_j^{(t)} = \text{softmax} \left(\text{score}(h_j^{(s)}, h_{t-1}^{(d)}) \right) \quad (4.29)$$

そして $\beta_j^{(t)}$ を用いて、 j 番目の節における i 番目の単語に対する Attention 分布 $\alpha_{(j,i)}^{(t)}$ を次より計算する。

$$\alpha_{(j,i)}^{(t)} = \text{softmax} \left(\beta_j^{(t)} \text{score}(h_{(j,i)}^{(e)}, h_{t-1}^{(d)}) \right) \quad (4.30)$$

最後に時間 t における Context ベクトル c_t を計算する。

$$c_t = \sum_{j=1}^N \sum_{i=1}^M \alpha_{(j,i)}^{(t)} h_{(j,i)}^{(e)} \quad (4.31)$$

最後の単語ごとの Attention 分布の計算において、節ごとの Attention 分布を重みとして与えることにより、節それぞれで注意すべき単語の情報を与えることが可能となっている。これにより、

通常の Pointer-generator Network の場合では、論文における序論の内容のみが要約として出力されるが、Discourse-Aware Attention Model では結論の内容まで出力される要約の内容に含まれるようになる。

しかし、このモデルは教師あり学習のニューラルネットワークであるため、モデルの構築にはより多くの学習データが必須であるため、先述したように現在公開されている日本語のコーパスでは十分な学習を行うことが不可能である。

2.4.6 Transformer

RNN や LSTM による Encoder-Decoder モデル、及びその発展モデルは文章生成において重要なモデルであるが、問題点として、計算コストがあった。再帰型ニューラルネットワークでは、時間方向にも計算するため、時間ごとの計算を並列化することができず、そのために、系列データが長いほどその計算コストがより大きくなる。そこで時間方向の計算を排した系列データを取り扱うニューラルネットワークモデルとして、Transformer が提案された。

Transformer は Pointer-generator Network と同様に Attention を採用した Encoder-Decoder モデルであるが、その構造は大きく異なる。Transformer の構造を図 4.7 に示す。右側が Encoder であり、左側が Decoder である。図よりわかる通り、RNN は使用せず、Attention のみを使用している。また、Encoder, Decoder の両方で Positional Encoding と呼ばれる埋め込みが入力の埋め込みに対して加算されているなど、RNN を用いた Encoder-Decoder モデルとは大きく異なる構造をしているのが確認できる。本節では順伝搬、Positional Encoding 及び、Transformer の Attention 機構を解説する。

Positional Encoding

これは、入力時に、データの位置情報を付加するために行う処理である。これは時刻 pos の入力データの埋め込みベクトルの i 次元の要素に対して以下の値 PE を加算する。

$$PE(pos, 2i) = \sin(pos/1000^{2i/d_{model}}) \quad (4.32)$$

$$PE(pos, 2i + 1) = \cos(pos/1000^{2i/d_{model}}) \quad (4.33)$$

ここで d_{model} はモデルに入力する際の埋め込みベクトルの次元数である。これを用いることで、入力データごとに位置情報を含ませ、同一単語でも類似した埋め込みベクトルにならないようにすることができる。

Attention

Transformer における Attention は Pointer-generator Network とは大きく異なる Attention を採用している。Pointer-generator Network における Attention の計算は式 4.16 で示されてい

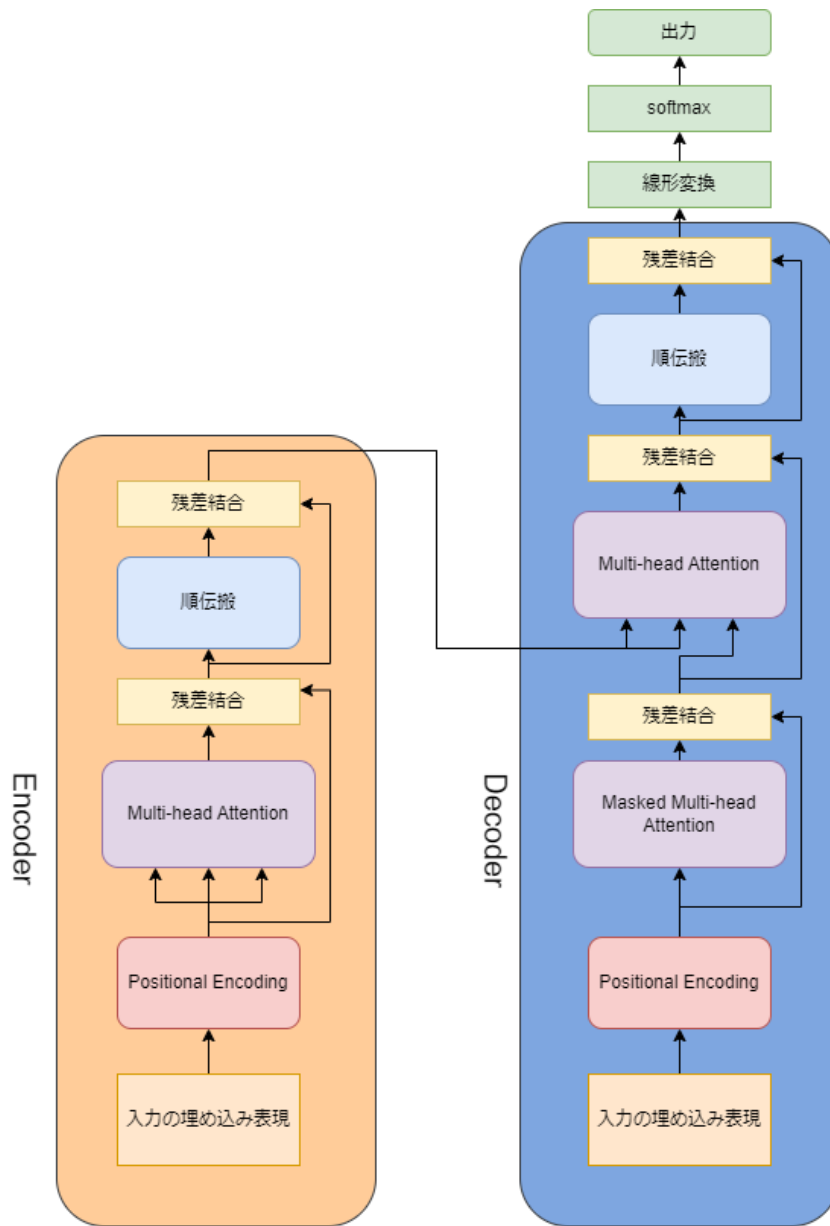


図 4.7 Transformer モデルの概略図. 実際には Encoder, Decoder それぞれで同じものを複数連結して構築する.

るが, Transformer では Encoder より得られる隠れ層の状態を *Key* (以下 K) 及び *Value* (以下 V) として扱い, Decoder の状態を *Query*(以下 Q) として, Attention の最終出力 $o(Q, K, V)$ を次の式から得る.

$$o(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (4.34)$$

ここの d_k は Q 及び K の次元数を表す. この Attention の計算を図 2.4.6 に示す. Pointer-

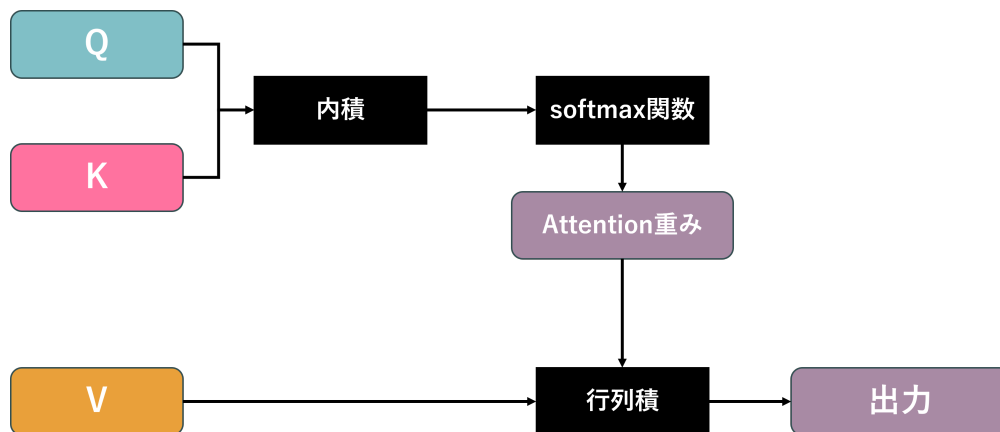


図 4.8 Multihead Attention の可視化. 複数の重みを用いて射影を生成し、それぞれで Attention を行う.

generator Networks では加法で Encoder の隠れ層の状態と Decoder の隠れ層の状態を結合し、重みを通して計算していたが、Transformer では内積を用いて求めている。これにより更新する必要があるパラメータの数を減らすことが可能となり、高速化できる。得られた最終出力は Encoder への入力 x との残差結合と正規化を行う。

また、Transformer では Multi-head Attention という Attention の並列化を行っている。これは入力データの埋め込みベクトルから得られる Q, K, V に対して、複数の重みを用いて、疑似的な射影を行い、Attention の計算をしてから結合するものである。これは次式のように表現できる。

$$Multihead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \quad \text{where } head_i = o(QW_i^Q, KW_i^K, VW_i^V) \quad (4.35)$$

ここで h は射影する数であり、モデルへの入力時の次元数に対して割り切れる数である必要がある。

Transformer では Attention を、self-Attention と Source-Target Attention の 2 つのパターンで行っている。self-Attention は Q, K, V の全てが同じ入力となっているもので、Encoder, Decoder の両方で使用されている。Decoder で使用する場合、学習時は全データをまとめて入力するという形をとる。しかし、その場合、予測すべき単語の情報を参照できてしまうという問題が起きるため、Attention における Q と K の内積の結果に対して、未来の情報に該当する要素を $-\inf$ としてマスキングする。生成などの推論時には Decoder 側の出力を再度 Decoder への入力として用いる自己回帰の形で行うため、問題はない。Source-Target Attention は Decoder 側のみで使用する。このとき、 Q, K は Encoder 側からの出力であり、 V は Decoder 側での self-Attention の出力である。

順伝搬

Multi-head Attention 及び, その残差結合を終えた後は最後に 2 回の線形変換を行う. ただし, 線形変換を行う間に ReLU 活性化関数を挟んでいる. ReLU 活性化関数 $Relu$ は次式で定義される.

$$Relu(x) = \max(0, x) \quad (4.36)$$

なので, 順伝搬における線形変換 FFN は, それぞれの重みを W_1, W_2 とし, バイアスを b_1, b_2 とすると, 次式で行われる.

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (4.37)$$

2.4.7 Transformer の発展モデル

Transformer の登場以降はこのモデルの構造を基本として, 複数の発展形が生まれたが, それらは凡そ以下の 3 種類に分類できる.

1. Encoder のみを使用したもの
2. Decoder のみを使用したもの
3. Encoder, Decoder の両方を使用したもの

それぞれについての代表的なモデルについて解説する.

Transformer-Encoder モデル: BERT

まず, 1 番目の Encoder のみを使用したモデルとして代表的なモデルは BERT である. こちらは単純に Transformer の Encoder をスタックしたものであり, Encoder モデル自体に大きな変更はない. ただし, 入力時の埋め込みベクトルに対して, Positional Encoding だけでなく, 新たに Segment Embedding と呼ばれる, 文の順番を示す埋め込み表現を加える. この Segment Embedding 自体には特に計算式はなく, 1 と 0 などの文の順番が明示されていれば埋め込み表現自体は任意のものを使用できる. 最終的には単語埋め込みと位置埋め込み, そしてセグメント埋め込みの 3 つの埋め込みの和を入力として BERT へと入力する. また, 入力する文の先頭には CLS トークン, 文同士の間には SEP トークンという特殊なトークンを挟む. これらの入力の構造を図 4.9 に示す.

BERT ではモデルの学習として Masked Language Model(MLM) 及び, Next Sentence Prediction(NSP) と呼ばれる, 2 つのタスクを解いて学習を行う. MLM は, 入力の一部単語をマスキングして, そのマスキングした単語を予測するというタスクである. ただし, 単純にランダムに単語をマスキングを行うのではなく, 次のルールに沿って単語ごとにマスキングする. 「少年が

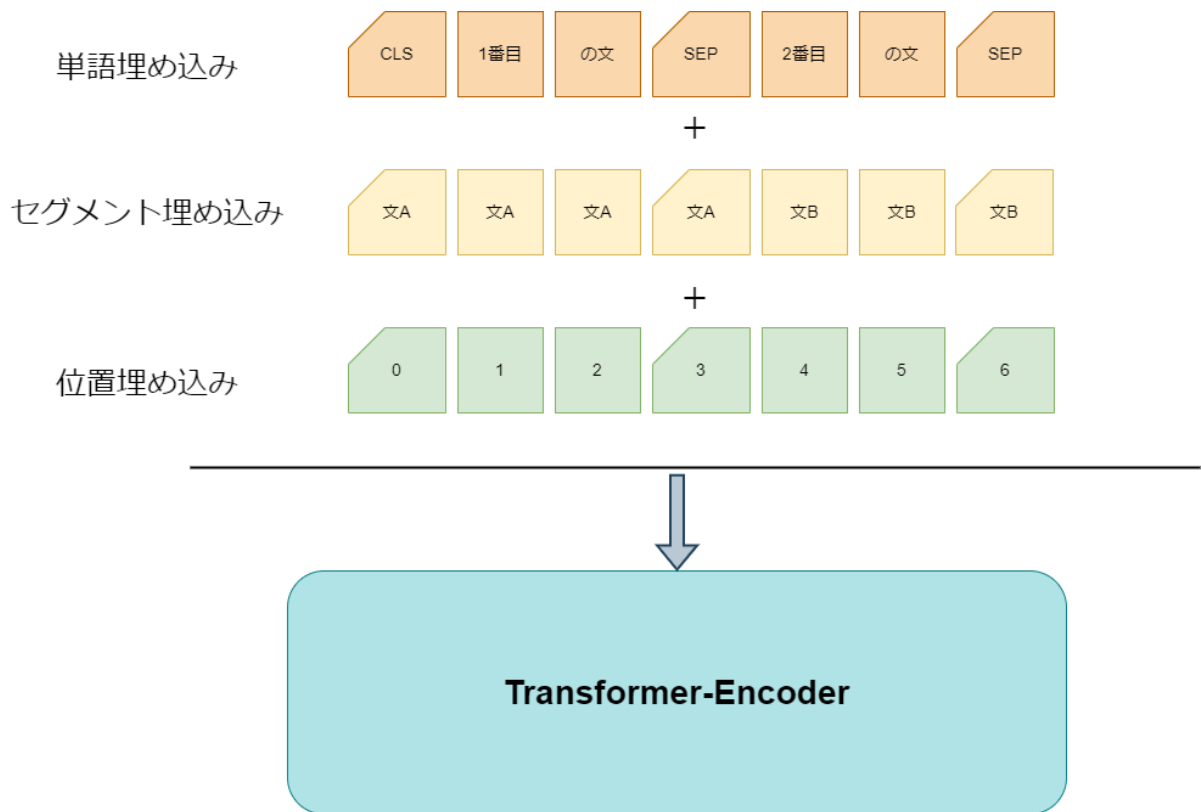


図 4.9 BERT への入力の流れ. BERT へ埋め込みを行う際には単語埋め込みと位置埋め込み (Positional Encoding) に加えて, セグメント埋め込みを加える. セグメント埋め込みによって, 文の区別をするための情報を与える.

山にある神社に向かった。」という文を例として, 「神社」という単語を対象にどのように置換するのか具体例も示す.

80% の確率でマスク 少年が山にある [MASK] に向かった.

10% の確率で別の単語に置換 少年が山にある寺に向かった.

10% の確率で何もしない 少年が山にある神社に向かった.

マスクされた箇所, もしくは置換された箇所に対応する出力が, 本来の正しい単語になるように MLM では学習を行う. 対して, NSP では, 与えられた 2 つの文章が連続している文章かどうかを判定するタスクである. 学習に使用するデータセットから入力する文を選択した後, データセット内から 50% の確率で実際に連続している文を選択し, 50% の確率で連続していない文を選択する. NSP の学習では CLS トークンに対応する出力を予測の判定に使用して学習する.

BERT は基本的に BERT 単体で用いるのではなく, 任意のタスクを解くためのモデルと結合し, BERT と結合したモデルを含めた全体で解きたいタスクのための学習を行う. このような,

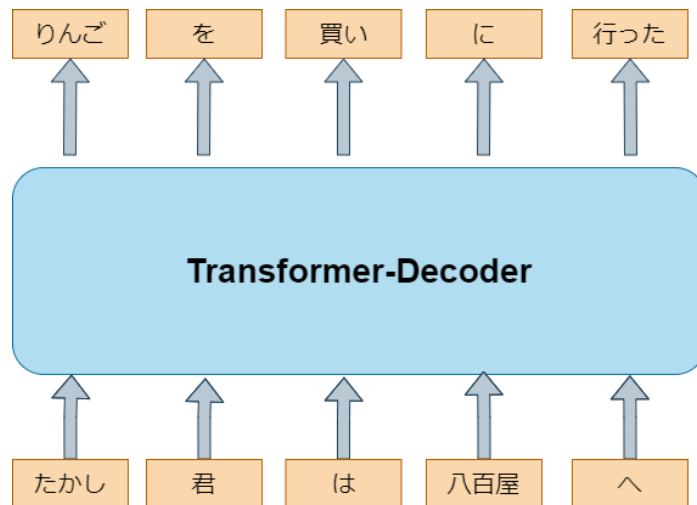


図 4.10 GPT における入出力の構造. 基本は Transformer の Decoder のみを使用して学習する. 例えば「たかし君が八百屋へ」という文までが入力された場合, Decoder からは「買いに行った」という文章が出力されるように学習する.

事前に学習したモデルに新規に構築したモデルを結合し, 解きたいタスクに対応する学習することを, ファインチューニングと呼ぶ. BERT を用いるファインチューニングにおいては, BERT より得られる, 入力単語に対応した出力以外にも, CLS トークンや SEP トークンも有用である. 例えば, CLS トークンに対応する BERT の出力は主に以下のようなファインチューニングに使用できる.

- 文章の分類
- ペア同士の文章の分類 (含意関係など)

実際, 先に紹介した BERTSum では文同士の SEP トークンを CLS トークンに変えたうえで, 抽出型要約モデルを BERT に接続し, CLS トークンを用いて抽出型要約を行うためにファインチューニングしているといえる.

Transformer-Decoder モデル: GPT

2 番目の Decoder のみを使用したモデルで有名なものは chatGPT に使用されている GPT モデル [26] が該当する. GPT では Transformer の Decoder を使用し, 事前学習として事前の k 個の単語を元に $k + 1$ 個目の単語を予測する, というものである. 例えば, 「たかし君が八百屋へ買いに行った」という文で, 入力として「たかし君が八百屋へ」が与えられたときに続きである「買いに行った」を出力するように学習する. この様子を図 4.10 に示す.

GPT ではこのような事前学習の後に分類タスクや生成タスクなどのためのファインチューニングを行うが, GPT-2[27] や GPT-3[28] では事前学習のみを行い, ファインチューニングは行わ

ず、そのままのモデルで運用する。そのため、特定のタスクを解く際には、一切学習をせずにそのままタスクを指示する、もしくは入力の中にタスクの解法例を含ませて、学習せずに入力するという方法を取る。例えば、和英翻訳というタスクを GPT-3 で行う場合、次の 2 通りの入力 GPT-3 においては可能である。

タスクのみを指示する場合の入力

「明日の朝食は目玉焼きがいい」という文を英語に翻訳せよ。

入力の中に解法の例を示して入力する場合

翻訳例:

朝食⇒ breakfast

目玉焼き⇒ sunny-side up

上の例を元に次の文を英語に翻訳せよ。

「明日の朝食は目玉焼きがいい」

これが実現できるのは、GPT-2 や GPT-3 では学習データ数及びモデルのパラメータ数が非常に大きいためである。GPT-3 では学習データセットに含まれる単語数は 4990 億、パラメータ数は最も大きなモデルで 1750 億ほどある。対して、BERT は大きなモデルでも単語数は 33 億、パラメータ数は 3 億 4 千万である。そして、chatGPT では人間による評価を用いた強化学習を行っているとされており、現在では自然言語処理における生成モデルとして代表的なモデルとなっている。chatGPT の詳細な情報は第 6 章にて述べる。

Transformer モデル: T5

3 番目の Encoder と Decoder の両方を用いたモデルとして代表的なものは T5[29] である。T5 とは Text-to-Text Transfer Transformer の略であり、その名の通り、テキストで入力してテキストで出力する Transformer である。そのため、モデル構造自体には大きな変化はなく、また事前学習自体は BERT と同様に MLM を採用している。BERT と異なる点として、単語単位で行うのではなく、複数の連続した単語を対象にマスキングをする。そして、マスキングがされた箇所の単語を、どの箇所かを明示した上で生成する。次にその例を示す。

T5 における事前学習タスクの例

元の文章: 少年が山にある神社に向かった。

マスキング例: <X>山にある <Y>。

出力例: <X>: 少年が <Y>: 神社に向かった

T5 では、様々なファインチューニング先のタスクにおいて入出力をテキストで行うためのフォー

マットを提案している。ファインチューニングを行う際には、タスクを表す Prefix を入力文の先頭に付与することで、マルチタスクの学習を行うことが可能である。例えば、次のように入力の先頭にタスク内容を端的に表す文言を付与する。

翻訳する場合 translate to English: 丘の向こうにある一軒家。

質疑応答する場合 answer this question: 日本の首都は何ですか。

要約をする場合 summarize: 彼は植物園へ行った。植物園でバラを見てきた。

含意関係を確認する場合 check the semantic relationship: sentence 1: トマトはナス科の植物である。sentence 2: ナスとトマトは同じ種類の植物である。

このように、Prefix を付与することで、異なるタスクでも、同じモデル、同じ損失関数で学習を行わせることが可能となり、パラメータを共有させることが可能となるという大きな利点がある。またこれまでの Encoder-Decoder モデルと同様に Prefix なしの学習も可能である。

mT5

T5 の多言語版対応版が mT5[30] である。T5 との違いは大きくなく、モデル自体の変更点として順伝搬時の処理に ReLU 活性化関数を用いていたのを、GeGLU 活性化関数に変更している。GeGLU とは活性化関数である Gaussian Error Linear Unit(以下 GeLU) と Gated Linear Units を組み合わせた活性化関数である。GeLU 活性化関数 $Gelu$ は次の式で定義される。

$$Gelu(x) = x\Phi(x) = x \cdot \frac{1}{2} \left[1 + erf(x/\sqrt{2}) \right] \quad (4.38)$$

ここで Φ は標準正規分布の累積分布関数、 erf は誤差関数である。誤差関数が解析的に計算することができないため、実際には次式で近似して計算する。

$$Gelu(x) \simeq 0.5x \left(1 + \tanh \left[\sqrt{2/\pi} (x + 0.044715x^3) \right] \right) \quad (4.39)$$

場合によってはシグモイド関数を用いて行う場合もあり、その場合は次式で近似する。

$$Gelu(x) \simeq x\sigma(1.702) \quad (4.40)$$

対して、Gated Linear Units(以下 GLU) 活性化関数 GLU は入力 x を 2 つに分け、2 つの異なる線形変換でのアダマール積を取る。それぞれの線形変換に用いる重みを W, V 、バイアスを b, c としたとき、 GLU は次のように計算する。

$$GLU = \sigma(xW + b) \otimes (xV + c) \quad (4.41)$$

GeGLU は GLU におけるシグモイド関数を GELU に変えたものである。Transformer における順伝搬の線形変換の式 4.37 より、mT5 における順伝搬の線形変換 FFN_{mT5} は次式を用いて行う。

$$FFN_{mT5} = (GELU(xW) \otimes xV)W_2 \quad (4.42)$$

事前学習モデル構築に使用しているのは日本語を含む多言語コーパスである mC4 である。このコーパスは Common Crawl から構成されている。Common Crawl は公開されている Web ページをクロールし、アーカイブしている非営利団体であり、アーカイブ化された Web ページは全て HTML 言語などのマークアップ言語や、テキストではない部分を削除して研究などに提供されている。Common Crawl は毎月 30 億から 50 億の新規ページをクロールしており、現在では 2400 億の Web ページがアーカイブ化されている。mC4 は以下の方針でコーパスを構築している。

- 読点や感嘆符、疑問符や引用符の最後などの区切り記号で終わる文章のみを使用
- 5 文末満で終わるページは無視し、最低でも 3 単語以上を含んでいる文章のみを使用
- 汚い言葉使いや不適切な単語を含むページを削除
- Javascript という単語を含む文を全て削除
- "lorem ipsum" という単語がプレースホルダーとして出てきたため、この単語を含むページを削除
- 中括弧を含んでいるページを削除
- 任意の 3 文が一度でも重複したデータを削除
- 言語判定には cld3[31] を使用し、107 言語からなるデータセットを構築
- 英語特有の記法は他言語では出現しないため、200 文字以上の文が最低でも 3 文あるページを使用
- cld3[31] による言語判定において、主要言語の信頼度が 70% 以下のものは削除

さらに学習を行う際には、学習リソースの少ない言語における過学習と、英語における学習不足を回避するために、リソース数に比例するサンプリング確率を設けることで、英語以外の言語でも英語と同様の性能を発揮することができる。

第3章

自動要約に対する評価手法

3.1 構成

本章は以下の構成からなる.

- 要約の評価と自動化
- ROUGE
- BERTScore
- QAEval

要約の評価と自動化では, 自動要約における評価の背景について述べる. ROUGE, BERTScore, QAEval のそれぞれは現在における自動要約における自動評価手法の代表的な手法である. 要約の自動化の背景について述べたのちに, これらの手法について解説する.

3.2 要約の評価と自動化

自動要約モデルによって生成された要約に対する評価の観点として, 文法的に正しいか, 文が流暢か, 必要な内容を網羅しているか, などが挙げられる. しかし, このような観点で評価を行うため, かつての自動要約モデルに対する評価の多くは人手によって行われていた [32, 33]. しかし, 人手による評価の場合, 以下の問題がある.

- 評価者を雇う必要があるため, その分費用が掛かる.
- 人手で行うため, 時間が掛かる
- 場合によっては身内で評価をするため, 再現性が低い

このような問題を避けるために, 自動要約モデルに対する評価の定量化, 及び自動化が必要であった. そこで Lin らは機械翻訳にて使用される自動評価手法である, BLEU[34] をベースとし

た評価手法が人手の評価と相関があることを示した [35].

BLEU はとある生成した翻訳文 C と正解データの翻訳文に含まれる n -gram に対する適合率を計算する. n -gram とは, とある文書から生成可能な n 個連続している単語の集まりのことである. たとえば, 次のような文 s があるとする.

s ある男の子が犬を連れて散歩をしている.

この文に対して n -gram を抽出するとき, $n = 1$ の場合を $s_{1\text{-gram}}$ とすると次の集まりが得られる.

$$s_{1\text{-gram}} = \{ \text{ある, 男の子, が, 犬, を, 連れて, 散歩, を, して, いる.} \} \quad (2.1)$$

$n = 1$ とした場合は単に単語の集まりとなる. 次に $n = 2$ としたとき, $s_{2\text{-gram}}$ は次のようになる.

$$s_{2\text{-gram}} = \{ (\text{ある, 男の子}), (\text{男の子, が}), (\text{が, 犬}), (\text{犬, を}), (\text{を, 連れて}), (\text{連れて, 散歩}), (\text{散歩, を}), (\text{を, して}), (\text{して, いる}) \} \quad (2.2)$$

このように n -gram では n 個連続している単語の組みを全て抽出する. この n -gram について BLEU では次式で予測率 p_n を計算する.

$$p_n = \frac{\sum_{n\text{-gram} \in C} \text{Count}_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in C} \text{Count}(n\text{-gram})} \quad (2.3)$$

$\text{Count}_{clip}(n\text{-gram})$ は正解データの翻訳文の n -gram と, 翻訳文 C の n -gram の間で一致したものの数を表す. $\text{Count}(n\text{-gram})$ は C 内の n -gram の数を表す. そして, BLEU ではこの適合率を元に, 生成された翻訳文に対するスコア $score$ を計算する.

$$score = \max(1, e^{1-r/c}) \cdot \exp \left(\sum_{n=1}^N w_n \log p_n \right) \quad (2.4)$$

ここで, r, c はそれぞれ正解データの翻訳と生成された翻訳の文の長さを表す. また, w_n は重みであり, $w_n = 1/N$ としている.

Liu らはこの BLEU のスコアの導出において, 適合率を用いるのではなく予測率を使用した. つまり, 正解データの翻訳文を R として, 式 2.3 を次のように変更した.

$$p_n = \frac{\sum_{n\text{-gram} \in C} \text{Count}_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in R} \text{Count}(n\text{-gram})} \quad (2.5)$$

式 2.5 を見てもらえばわかる通り, 分母が正解データの翻訳文の n -gram の数になっているため, 予測率の計算となっている. Lin らは予測率にさらに生成した要約が短い場合に対してボーナスを付与することを提案していたが, 実験においてはボーナスを使用していない.

この n -gram の完全一致による自動評価が人間による評価との相関があることが示されたため, n -gram 一致による要約の自動評価として ROUGE[36] が提案された.

3.3 ROUGE

ROUGE は n -gram 一致による要約の自動評価パッケージである。パッケージであるため、複数の評価方法が提案されており、現在でも使用されている代表的な評価指標である。

n -gram 一致による評価方法ではあるが、その一致の条件、そして一致した際の評価の仕方も異なっている。以下に、ROUGE に含まれる代表的な評価方法の一覧とその概要を示す。

表 3.1 ROUGE に含まれる代表的な評価手法の一覧

名前	概要
ROUGE-N	n -gram 一致による評価。単純に一致していれば評価は上がっていく。
ROUGE-L	どれだけ単語の出現順が一致しているかどうかで評価する。
ROUGE-S	単純な単語の組み合わせで評価。

本節では ROUGE にて提案されている各評価指標について解説する。

3.3.1 ROUGE-N

ROUGE-N は先の Liu らと同様に任意の n -gram に対する予測率を計算する。しかし、異なる点として BLEU、及び Liu らの手法では複数の n -gram について適合率もしくは予測率を計算し、その対数の総和を用いて計算していたが、ROUGE-N では一つの n -gram のみで計算を行う。ROUGE-N におけるスコア $score_{rouge-n}$ の計算式は以下の通りである。

$$score_{rouge-n} = \frac{\sum_{gram_n \in R} Count_{match}(gram_n)}{\sum_{gram_n \in R} Count(gram_n)} \quad (3.1)$$

$n = 1$ の場合は ROUGE-1、 $n = 2$ の場合では ROUGE-2 と呼ぶときもある。

例えば、次のような、生成された要約と正解の要約のペアがあったとする。

生成された要約 少年が犬と一緒に公園へ行った。

正解の要約 近所の男の子が犬を連れて公園へ行った。

これに対して ROUGE-1 で評価を行う場合、まず、これらを単語に分割する。以下に、上記 2 つの文を単語に分割したものを示す。

生成要約の単語

少年, が, 犬, と, 一緒, に, 公園, へ, 行っ, た

正解要約要約の単語

近所, の, 男の子, が, 犬, を, 連れ, て, 公園, へ, 行っ, た

このとき, 共通している単語の数は 5 単語なので, 式 3.1 より,

$$score_{rouge-1} = \frac{5}{10} = 0.5$$

スコアは 0.5 となる.

3.3.2 ROUGE-L

ROUGE-L では, Long Common Sequences(以下 LCS) と呼ばれる, 正解となる要約文と生成した要約文の間で, 順序を含めて一致している箇所を基準として評価を行う. 例えば, 次の 2 文 X, Y があったとする.

X 明日は公園に行って友達とサッカーで遊ぶ予定だ.

Y 公園でサッカーで遊ぶつもりだ.

このとき, X と Y の LCS は「公園, サッカー, 遊ぶ, だ」の 4 単語となる. 対して新たに文 Z があったとして, その内容が以下の通りであったとする.

Z サッカーで遊びたい. 公園で.

このときの X と Z の LCS は「サッカー, で」の 2 単語となる. このように順序が一致していない場合には同じ単語を使用している場合でも, LCS にはカウントされない. 逆に単語とその順序が一致している場合には, 間にどんな単語が挟まれていても無視することができる.

この LCS について, ROUGE-L では予測率及び適合率を計算し, その調和平均をスコアとする. まず, 正解要約 C と生成要約 G に対する, ROUGE-L の予測率 R と適合率 P は次式より計算する.

$$R = \frac{LCS(C, G)}{M} \quad (3.2)$$

$$P = \frac{LCS(C, G)}{N} \quad (3.3)$$

このとき, M, N はそれぞれ C, G それぞれの単語数を表す. そしてこれらの調和平均をスコア $score_{rouge-l}$ を計算する.

$$score_{rouge-l} = \frac{2RP}{R+P} \quad (3.4)$$

例えば先の文 X を正解要約, Y を生成要約として ROUGE-L による評価を行うとする. このとき, X と Y の LCS は先に示してある通りなので, X と Y の LCS の単語の数は 4 単語. なの

で, X と Y の ROUGE-L のスコアは式 3.2 より計算する.

$$\begin{aligned}R &= \frac{4}{13} \sim 0.30 \\P &= \frac{4}{7} \sim 0.57 \\score_{rouge-l} &= \frac{2 \times \frac{16}{91}}{\frac{80}{91}} \\&= \frac{32}{80} = 0.40\end{aligned}$$

となり, 最終的にはスコアは 0.40 となる.

ROUGE-L では LCS を用いて計算するが, 先述したように間にある単語は無視するため, 「公園に行ってサッカーと野球で遊ぶ予定だ」のような文章でも同一のスコアが与えられる. そのため, この問題を回避するため, 連続一致している場合にはより高い評価を与える ROUGE-W という手法も提案されている. ROUGE-W では, スコアを記録する 2 次元テーブル c と, 連続一致した際の長さ k を記録する 2 次元テーブル w を用いて重みづけされた LCS を計算をする. ROUGE-W における LCS の重みづけはアルゴリズム 3 で行われる.

Algorithm 3: ROUGE-W の LCS の重みづけアルゴリズム

```
c(i, j); // スコアの格納テーブル
w(i, j); // 連続一致した際の長さを格納するテーブル
  for i = 0, i ≤ M do
    for j = 0, j ≤ N do
      c(i, j) ← 0;
      w(i, j) ← 0;
    end
  end
  for i = 1, i ≤ M do
    for j = 1, j ≤ N do
      if xi = yj then
        // 単語が一致したとき
        k = w(i − 1, j − 1); // 直前までの連続一致した際の長さを取り出す.
        c(i, j) = c(i − 1, j − 1) + f(k) − f(k − 1); // スコアを記録
        k = k + 1
      end
      else
        if c(i − 1, j) > c(i, j − 1) then
          c(i, j) = c(i − 1, j);
          w(i, j) = 0;
        end
        else if c(i, j) = c(i, j − 1) then
          w(i, j) = 0;
        end
      end
    end
  end
end
return c(M, N)
```

アルゴリズム 3 中にある f は連続一致した際の長さに基づく重みづけ関数であるが、次の条件を満たす必要がある。

- 与えられる長さ k は常に $k > 0$
- 任意の正の整数 x, y が与えられたとき、 $f(x + y) > f(x) + f(y)$

また、スコアの計算は予測率と適合率の調和平均である点は ROUGE-L と同じだが、予測率と適合率の計算が異なる。これらの計算は式 3.5 によって行われる。

$$\begin{aligned} R &= f^{-1} \left(\frac{W LCS(X, Y)}{f(M)} \right) \\ P &= f^{-1} \left(\frac{W LCS(X, Y)}{f(N)} \right) \end{aligned} \tag{3.5}$$

評価を行う際には重みづけに使用した関数の逆関数を使用する。

3.3.3 ROUGE-S

ROUGE-S では単純な単語の組み合わせの一致で評価を行う。ただし、単純に組み合わせを行うのではなく、組み合わせができる範囲を限定する。例えば、ROUGE-S4 の場合、ある単語から先の 4 単語の範囲にある単語でのみ組み合わせが作成が可能である。ROUGE-S でも、ROUGE-L と同様に予測率 R と適合率 P の調和平均をスコアとして使用する。ROUGE-S における予測率と適合率の計算は次式から行う。

$$R = \frac{SKIP2(C, G)}{Comb(C)} \tag{3.6}$$

$$P = \frac{SKIP2(C, G)}{Comb(G)} \tag{3.7}$$

ここで、 $SKIP2$ は正解要約と生成要約の間で一致した単語の組み合わせの数、 $Comb$ は要約文より得られる単語の組み合わせの数を示している。調和平均の計算は式 3.4 と同じである。

ここで ROUGE-N において、例として用いた 2 文を使って ROUGE-S における評価を行う。今回は単語の組にできる範囲を 2 単語として行う、ROUGE-S2 として評価する。先の 2 文は以下のような文章である。

生成された要約 少年が犬と一緒に公園へ行った。

正解の要約 近所の男の子が犬を連れて公園へ行った。

これらの文章内に含まれる単語の組を、2 単語までの範囲で作成すると以下のような組み合わせが作成できる。

生成要約からできる単語の組

(少年, が), (少年, 犬), (が, 犬), (が, と), (犬, と), (犬, 一緒), (一緒, に), (一緒, 公園), (に, 公園), (に, へ), (公園, へ), (公園, 行っ), (へ, 行っ), (へ, た), (行っ, た)

生成要約からできる単語の組

(近所, の), (近所, 男の子), (の, 男の子), (の, が), (男の子, が), (男の子, 犬), (が, 犬), (が, を), (犬, を), (犬, 連れ), (を, 連れ), (を, て), (連れ, て), (て, 公園), (て, へ), (公園, へ), (公園, 行っ), (へ, 行っ), (へ, た), (行っ, た)

これらの組み合わせの中で一致する組の数は5組なので, スコア $score_{rouge-s}$ は式 3.6 より,

$$\begin{aligned}R &= \frac{5}{15} \sim 0.33 \\P &= \frac{5}{20} = 0.4 \\score_{rouge-s} &= \frac{2 \times \frac{25}{300}}{\frac{35}{60}} \\&= \frac{5}{35} = \frac{1}{7} \sim 0.14\end{aligned}$$

なので, スコアは 0.14 として評価される.

ROUGE-S では, さらに 1-gram の一致も評価に加えた ROUGE-SU という評価手法も提案されている. 上記の例の場合, さらに 1-gram の一致が加わるので, 一致しているのは 5組 +5 単語なので, ROUGE-SU によるスコア $score_{rouge-su}$ は,

$$\begin{aligned}R &= \frac{10}{20} = 0.5 \\P &= \frac{15}{25} = 0.6 \\score_{rouge-su} &= \frac{2 \times 0.5 \times 0.6}{0.5 + 0.6} \\&= \frac{0.6}{1.1} \sim 0.54\end{aligned}$$

と計算されるため, 0.54 と評価される.

3.3.4 ROUGE の問題点

ROUGE における評価における問題点として, 言い換えへの対応ができないという問題点がある. 例えば, 次の 2 文があったとする.

文 A 彼は長時間, 店の前で待っていた.

文 B 彼は長い間, 店前で待機していた.

この 2 文は一見すると類似した意味の文章同士であるが, これを ROUGE-1 で評価する場合, 一致している箇所というのは名詞のみで考えると「彼」のみとなる. これは, 他の ROUGE-L や ROUGE-S でも同様であり, 「長時間」と「長い間」, 「店の前」と「店前」, 「待機」と「待っ

て」のような言い換えは同じ意味でも、字面が異なるため、一致したものと判定されず、評価されない。

この問題に対応した類似手法としては METEOR[37] がある。これは ROUGE-1 と同様に単語の一致による評価を行うが、この一致には言い換えや、類似語による意味的な一致も評価に加えるという特徴がある。しかし、この意味的な一致の判定には WordNet[38] を用いているため、評価が可能な言語が、特定の言語のみに特定されてしまうという問題がある。この言い換えにおける言語的問題を解決したのが、BERTScore[39] である。

3.4 BERTScore

3.4.1 BERTScore による評価

BERTScore は、その名の通り BERT を用いた評価手法である。この手法では、BERT より得られる単語の埋め込み表現を用いて、その類似度から評価を行う手法である。この手法の利点として、BERT を使用することで言い換えや類似語に対応するだけでなく、前後の文脈を考慮した埋め込み表現を生成することが可能なため、内容を考慮した評価を行うことが可能である。

BERTScore では次のように評価される。ある正解要約 C と生成要約 G があったとき、それぞれが $C = \{c_1, \dots, c_N\}, G = \{g_1, \dots, g_M\}$, のように単語の系列からなっているとす。このとき、それぞれの要約の単語の埋め込み表現 \hat{c}_i, \hat{g}_j を用いて、予測率 R と適合率 P を計算し、それらの調和平均 F を評価のスコアとする。

$$R = \frac{1}{|C|} \sum_{c_i \in C} \max_{g_j \in G} c_i^\top g_j \quad (4.1)$$

$$P = \frac{1}{|G|} \sum_{g_j \in G} \max_{c_i \in C} c_i^\top g_j \quad (4.2)$$

$$F = 2 \frac{RP}{R + P} \quad (4.3)$$

つまり、BERTScore では最も意味的に類似している単語を探し出し、それらの類似度の平均を評価として用いる。これを図 4.1 に可視化する。

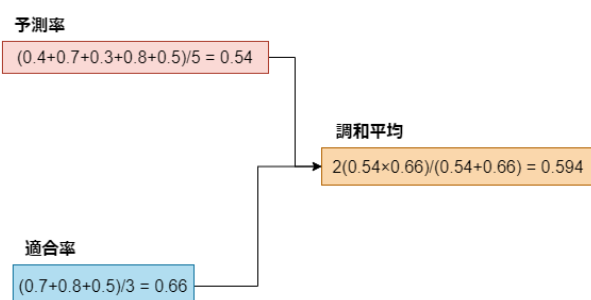
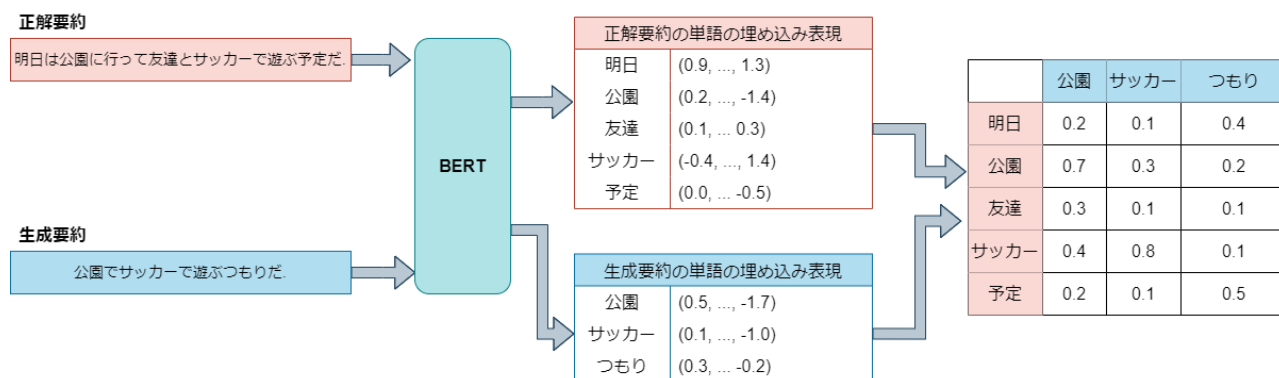


図 4.1 BERTSum における評価の流れ. それぞれの埋め込み表現を獲得した後, 内積で類似度を計算する. そして, 予測率と適合率を類似度を用いて計算して, それらの調和平均で生成した要約を評価する. 予測率は正解要約中の単語に対する類似度の最大値の平均値, 適合率は生成要約中の単語に対する類似度の最大値の平均値を計算する.

これにより, 意味的類似度を考慮した評価を行うことが可能だが, 単語の 1 対 1 対応でしか評価できない. この問題を解決した発展手法が MoverScore[40] である.

3.5 QAEval

3.5.1 概要

ROUGE 及び, BERTScore では基本的に単語の一致, 及び意味的類似度を基準として自動要約モデルに対する評価を行っている. このような評価に対して, あくまで要約同士のトピックについての類似度を評価しているのみであり, 同じ情報を持っているかどうかについては評価できていない, という指摘がある [41]. そのため, 正解要約と生成要約間で同じ情報を持っているかを評価する手法として, QAEval[42] が提案された.

QAEval では次の手順で評価を行う.

1. 正解要約から名詞句を抽出する

2. 正解要約を情報源として、抽出した名詞句が正答となるような質問を生成.
3. 生成要約を情報源として、生成した質問に回答
4. 回答が正答とあうかどうか評価し、この評価結果で要約を評価

以降の節にて、各手順の詳細について述べる.

3.5.2 名詞句の抽出

QAEval では先に、正解要約から可能な限り、名詞句を抽出する. 例えば次のような文章があったとする.

例 さとる君は白いシャツと青い半ズボンを穿いて、カブトムシを取りに近くの林へ出かけた.

このとき抽出可能な名詞句として「さとる君」、「白いシャツ」、「青い半ズボン」、「カブトムシ」、「近くの林」が挙げられる. 1つの要約から抽出可能な名詞句の数は、要約の内容や抽出に使用するツールに依存するが、QAEval では凡そ 110 個ほどとされている. 名詞句の抽出には様々な手法が存在するが、日本語においては Ginza による名詞句の自動抽出が可能となっている.

3.5.3 質問の生成と回答

正解要約から抽出した名詞句を正答とし、評価に使用する質問を生成する. 質問を生成するための情報源は正解要約を使用し、正解要約と同様の内容であれば、正答できるような質問を生成する. 質問の生成には質問生成モデルを使用する.

例えば、先の例文と抽出した名詞句を使用して質問を生成する場合、次のような複数の質問が生成される.

- カブトムシを取りにいったのは誰？ (正答: さとる君)
- さとる君はどんなシャツを着ていた？ (正答: 白いシャツ)
- さとる君の穿いていたズボンはどんなズボン？ (正答: 青いズボン)
- さとる君は林に何を取りにいった？ (正答: カブトムシ)
- さとる君はカブトムシを取りにどこへ行った？ (正答: 近くの林)

このように、抽出された名詞句に対応する質問を要約の内容に基づいて生成する. また、抽出された名詞句に対応する質問を生成するため、生成される質問の数は抽出された名詞句の数と同じになる.

3.5.4 回答に対する評価

生成した質問に対し、生成した要約を情報源として、質疑応答モデルで解答する。例えば、次のような文章があったとする。

例 さとるはシャツと半ズボンを着て、林にカブトムシを取りに行った。

これを使って先の例で挙げた質問の一つ、「カブトムシを取りにいったのは誰？」という質問に対して、質疑応答モデルが「さとる」と答えたとする。このとき、解答に対して正答かどうかを判定する方法は2通りある。一つは完全一致による評価と、予測率と適合率の調和平均による評価である。前者の場合、質問の正答は「さとる君」であるので、一致しないため0と評価する。逆に一致していた場合には1と評価する。後者の場合、一致している単語は「さとる」であるため、予測率は0.5、適合率は1として、調和平均の値は $2(0.5 \times 1)/(1 + 0.5) = 1.0/1.5 = 2/3 \sim 0.67$ となる。このような評価を全ての質問に対して行い、その総和を生成要約に対する評価とする。この評価の違いを図5.1に示す。完全一致の場合、一致しているのは4番目の質問のみであるので、最終的なスコアは1となるが、調和平均の場合、単語が一部一致していればよいので、最終的なスコアは完全一致よりも高くなる。

3.5.5 要約評価の指標について

要約評価における指標というのはこれまで述べてきたように、複数存在しているが、必ずしも一つの評価指標のみですべて評価できるとは限らない。QAEvalにおいて述べたように、BERTScoreのような一致ベースの手法では先述したように、正解の要約と生成した要約の間で同じ話題を取り扱っているかどうかは判定できるが、同じ情報を持っているかどうかについては評価できない。逆にQAEvalのような質疑応答ベースの場合は、同じ情報を持っているかどうかは判定できるが、同じ話題を取り扱っているかどうかの評価を行うことはできない。そのため、要約生成モデルに対する評価を行う際には複数の評価指標を用いて行うのが好ましいとされる。

質問	正答	生成要約から得られた解答
カブトムシを取りにいったのは誰？	さとる君	さとる
さとる君はどんなシャツを着ていた？	白いシャツ	シャツ
さとる君の穿いていたズボンはどうなズボン？	青いズボン	半ズボン
さとる君は林に何を取りにいった？	カブトムシ	カブトムシ
さとる君はカブトムシを取りにどこへ行った？	近くの林	林

完全一致の場合

$$0+0+0+1+0 = 1$$

調和平均の場合

$$0.67+0.67+0.5+1+0.5 = 3.34$$

図 5.1 QAEval における質問に対する解答の評価の違い。完全一致の場合は、一単語でも抜けていると間違いとして判定されるが、調和平均の場合は部分一致で評価を行うため、比較的完全一致よりも値は高くなりやすい。

第4章

日本語における自動要約性能の評価

4.1 本章について

第2章において、複数の抽出型要約、及び抽象型要約について述べた。これらの手法の多くは、英語で記述されたコーパスによって性能が評価されており、日本語などの英語以外の他言語における要約性能については未検証のままである。英語以外の言語において要約性能についての研究として、トルコ語における LSA を用いたトピックベースの抽出型要約の性能の検証がされた [43]。この検証では、英語の科学論文記事データセットである Summac データセットと、トルコ語で記述された科学論文のそれぞれで LSA によるトピックベースの抽出型要約の性能を比較したところ、英語と同様の性能を発揮することが示された。

また、抽象型要約においては、Transformer の登場以降、多言語事前学習モデルが登場しており、英語以外の言語でも利用することが可能となっているが、公開されている抽象型要約に利用可能な事前学習多言語モデルは現状、T5 の多言語モデルである mT5 のみである。また、検証は日本語を含めた複数の言語で行われたが、学習も複数の言語で行われていたため、日本語のみで学習した際の性能検証は行われていない。

そこで、まず、既存の抽出型要約及び抽象型要約が日本語の文書においてどれほどの性能を発揮するか、検証を行った。

4.2 LSA による抽出型要約手法の評価

先述の通り、抽出型要約において、LSA による抽出型要約が英語以外の言語において、同様の性能を発揮することが示されている。そこで、本研究でも同様に LSA による抽出型要約を用いて、日本語においても英語と同様に性能を発揮するかどうかを検証した。検証には [43] における英語の評価結果と比較をし、同様の評価結果となるかを確認した。抽出する文の数は文書内の文の数の 1 割までとし、評価方法は比較対象と同様に ROUGE-L を用いた。

4.2.1 使用コーパス

検証には自然言語処理学会論文誌コーパス [6] を使用した。このコーパスは 2013 年から公開されているコーパスで、言語処理学会論文誌である「自然言語処理」に掲載された論文が含まれている。最終更新は 2020 年のもので、3 月 16 日時点で収録件数は 629 件である。本研究では、この 2020 年 3 月 16 日時点のものを使用した。

この自然言語処理学会論文誌コーパスに収録されているデータは全て LaTeX 形式である。加えて、論文本体に関する LaTeX のソースコードだけではなく、論文掲載に必要な装飾用の LaTeX コードも多く記されている。そのため、本研究では学習用に独自のパーサーを作成し、加工を行った。以下にパーサーのルールを明記する。なお、パーサーの作成はプログラミング言語である Python を使用した。

- `\begin{document}` と `\end{document}` で囲われている箇所以外は論文本体ではないと判断し、全て削除。ただし、`\title`、`\Abstract` などの要旨や著者、タイトルを含む論文のメタデータは残す。
- `\begin{}` から `\end{}` で囲われている、`document` を除いた環境は内部を含めて削除。
- `\section{}` のみを残し、`\subsection{}` 及び、`\subsubsection{}` は題名含めて削除。
- 文字装飾に関するコマンド (例: `\bf`) は全て削除。
- 文章内部に記述されている数式は削除
- 要旨は、論文の LaTeX ソースに含まれている `\jabstract` 環境内部に記述されているものを採用。
- 論文本文の句読点は全て、半角ピリオド、及び半角カンマに変換し、統一。
- `\Appendix` 以下の文は全て付録とみなし、記録はするが学習には使用しない。

加工後は、json ファイルとして記録した。ソースコード 4.2.1-1 に json ファイル記録時の形式を掲載する。

ソースコード 4.2.1-1 加工後の json ファイルの保存形式

```
1  "ID"{
2    "title": 論文のタイトル,
3    "Abstract": 論文の要旨,
4    "Main": {節の題名
5      : 本文,
6      ...,
7    },
8    "Appendix":
9  }
```

なお、ソースコード 4.2.1-1 に記載されている、ID は自然言語処理学会論文誌コーパス内で割り振られた ID であり、「V[xx]N[yy]-[zz]」として与えられる。xx, yy, zz は以下の内容を表す。

- xx 該当の論文が掲載されている論文誌の巻の番号
- yy 該当の論文が掲載されている論文誌の号の番号
- zz 該当の論文が掲載されている論文誌における掲載順

この実験では上記の方法で抽出できた、要旨と本文が共に日本語で記述されている 429 件を評価に使用した。

4.2.2 結果と考察

英語と日本語それぞれにおける LSA による抽出型要約の評価結果を表に示す。

表 2.1 LSA による抽出型要約に対する英語と日本語それぞれの評価結果

	英語	日本語
ROUGE-L	0.180	0.181

日本語においても英語と同様の評価結果になっていることが確認できた。LSA による抽出型要約においては言語を問わず、性能を発揮することが示された。日本語においても同様の性能を発揮できた要因として、LSA では単語の並び順を考慮しないため、単語の共起性のみでトピックを抽出するために、英語と日本語における大きな差異である語順の違いを気にすることなくトピックを抽出することができたためだと思われる。

4.3 LDA への置換に伴う性能の変化の評価

4.3.1 概要

LSA による抽出型要約は、英語を対象とした場合と同様の性能を発揮することが確認された。さらに別の特徴量を用いて日本語文書の抽出型要約を行っても、性能を発揮することが可能なのか検証を行った。使用する特徴量として、LSA と同じ潜在的意味解析手法である LDA を使用した。

検証には LSA による抽出型要約での検証と同様に、自然言語処理学会論文誌コーパスを使用し、評価も同様に ROUGE-L による評価を行った。LDA のトピック数は 40 とした。

4.3.2 結果と考察

評価の結果を表 3.1 に示す。結果より、LDA に特徴量を変えても LSA の場合よりも性能は下

表 3.1 LSA と LDA それぞれによる抽出型要約の評価の結果

	LSA	LDA
ROUGE-L	0.181	0.197

がらず、性能が上昇していることが確認できた。上昇しているのは、LDA が確率分布を用いており、トピックを多項分布として表現されているため、複数のトピックについて考慮した特徴量を生成しているためであると考えられる。また、この結果から語順を考慮しない場合ならば日本語文書を対象にした自動要約では英語でしか検証されていない手法であっても有効である可能性がある。

4.4 mT5 による抽象型要約手法の評価

抽出型要約が日本語において性能を発揮するのかどうかを検証し、英語と同様の性能を発揮することを確認した。次に、抽象型要約における日本語での性能を評価する。評価方法として、抽象型要約にも使用できる多言語事前学習モデルである mT5 を用いて、日本語コーパスのみのファインチューニング前後での評価の変化を比較し、ファインチューニング後で評価が上昇するかどうか確認した。使用した mT5 は、日本語を含む 44 の言語から構成される多言語要約用コーパス XLSum[44] を用いて追加で学習されたモデルを使用した。評価方法は抽出型要約における実験と同様に ROUGE-L を用いた。

4.4.1 使用コーパス

有用性の検証には毎日新聞社速報記事データ 2018 年版 [45] を使用した。このデータセットは毎日新聞社から購入可能なデータセットであり、ポータルサイトなどにて配信された速報記事についてのデータセットであり、2013 年版から 2021 年版までが存在する。速報記事にはそれぞれ要約が付与されており、それぞれの記事には要約として、150 文字ニュース、54 文字ニュースのいずれか、もしくはその両方が付与されている。それぞれの要約には以下の性質がある。

150 文字ニュース ニュース記事の冒頭 2, 3 文を抽出した要約

54 文字ニュース ニュース記事全体の要約

この性質のため、150 文字ニュースは抽象型要約としての性能を検証するには不適切である。そ

のため、この検証では 54 文字ニュースが付与されている記事のみを使用した。54 文字ニュースが付与されている記事は 2018 年版の場合、14422 件であったので、検証では、mT5 による要約生成のための学習データ数は 11537 件、評価用データは 2885 件とした。評価手法は ROUGE-L を用いて行い、ファインチューニング前後それぞれで要約を生成し、それらに対する ROUGE-L の評価を比較した。なお、mT5 のファインチューニング時の設定は以下に示す。

ファインチューニング時の設定

学習データ数 11537
バッチ数 16
エポック数 8
入力可能最大長 256 トークン
出力可能最大長 64 トークン

4.4.2 結果と考察

評価の結果を表 4.1 に示す。

表 4.1 mT5 の日本語要約生成における有効性の検証。ファインチューニング前と後でどれだけ性能が向上するかを比較した。

	ファインチューニング前	ファインチューニング後
ROUGE-L	0.319	0.556

ファインチューニング前後で ROUGE-L スコアが 0.237 上がっているのが確認できた。このことから、mT5 における日本語の抽象型要約はファインチューニングによって十分な性能を発揮することが確認できた。mT5 における事前学習及び、XLSum による学習において言語間での偏りが起きないように学習をしていたことに加えて、日本語以外のデータについても学習することで他の言語での学習時のパラメータを利用することが可能となっていたことが今回のファインチューニングによる性能上昇が大きかった原因であると思われる。また、このことから言語ごとに異なる語順の影響をファインチューニングによって解消し、多言語モデルの性能改善ができると判断できる。

4.5 本章のまとめ

本章では、抽出型要約と抽象型要約のそれぞれにおいて、日本語に対する要約の性能を評価した。抽出型要約では LSA によるトピックベースの抽出型要約にて、英語における LSA によるトピックベースの抽出型要約と同様の性能を発揮することを確認した。抽象型要約では多言語版事

前学習モデルである mT5 を用いて、日本語コーパスを用いたファインチューニングの前後での性能の変化を比較した。結果、ファインチューニングにより性能が大きく上昇し、日本語コーパスのみでも抽象型要約が可能であることが確認できた。

つまり、語順を考慮しないトピックベースの抽出型要約、及び日本語を含んだ多言語版の事前学習モデルによる抽象型要約は日本語における自動要約において有効であることが示された。この結果から、日本語で記述された論文要約においてこれらの自動要約手法は有効であると思われる。

次章からは、論文を対象とした自動要約について、本研究の提案手法について述べ、有効性検証についての実験について説明する。実験では、提案手法で用いる自動要約手法のベースとして、本章で日本語での有効性が確認できた手法を用いる。

第5章

文章構造に着目した論文要約手法

5.1 既存の論文の自動要約手法について

本研究の提案手法について述べる前に、論文を対象とした自動要約である、第2章で述べた Discourse-aware Attention Model について改めて確認する。Arman らの提案した Discourse-aware Attention Model は、節の埋め込み表現を得るための Encoder である RNN_{sec} と、節の埋め込み表現より論文の埋め込み表現を得るための Encoder である RNN_{doc} という2つの Encoder から構築されている。

Arman らの手法では、論文の構造を直接 Encoder-Decoder モデルに落とし込むことで、RNN などの再帰型ニューラルネットワークを用いる Decoder に対して節ごとの情報を伝達し、Decoder における出力に必要な情報を保持することができている。要旨の長さはニュース記事の要約と比較して、非常に長い。Encoder-Decoder モデルにおいて Decoder からの生成では、出力が長くなるほど先の出力に関係する隠れ層の状態というのは消失していく。そのため、Decoder へ最初に入力する Encoder 側からの隠れ層の状態というのは長い文章を生成する際には、最終的には消失してしまう。この問題を、Decoder での生成時に節ごとの Attention と単語ごとの Attention という形で論文本体の情報を与えることで解決した。しかし、この手法の場合、論文の要約を生成するために情報を与える際に、必要のない情報を与えている場合もある。例えば、関連研究の内容は論文を読み解くうえでは重要だが、要旨を作成する際には必要がない場合がある。

これに対する解決方法として、全ての節を要約の生成において使用するのではなく、要約を生成する前に要約に用いる節を選択することが考えられる。そして、選択する節は要旨の内容を含んでいる節であることが好ましい。しかし、どのような節が要旨の内容を含んでいるのかは明確ではない。なので、まず、論文の構造と要旨の構造について述べる。

5.2 論文と要旨の構造

5.2.1 論文の構造

論文の形式は基本的に投稿先となる学会、もしくはジャーナルによってページ数、フォントの種類及びサイズ、表示形式、含むべき内容というのは指定されている。例えば、以下のような指示がある。

要旨に対する指示 どのような形式で論文の内容をまとめるべきかの指示が書かれている。例えば参考文献への参照がないことや、文字数の指定がある。

内容に対する指示 研究の背景を含めることなど、アカデミックな研究としての書き方を行うべきという指示が書かれていることが多いが、実際にどのような内容を含むべきなのか、明確な指示があることは少ない。

図表に対する指示 載せる図表に対する解像度の指示など。場合によっては学会誌掲載における図表の取り扱い(出力する際の色など)についての注意書きが書かれている場合もある。

場合によっては学会側からテンプレートの使用を義務付けられていることもあり、表示形式の指示については具体的な指示をされることが多い。対して、上述したように内容に対しては、具体的な論文の構造や含むべき内容についての指示は少ない。しかし、多くの論文において以下の構造で執筆されていることが多い。

1. 研究背景
2. 研究目的
3. 関連研究
4. 提案手法
5. 実験
6. 考察
7. 結論

ただし、必ずしもすべての論文が同じ構造になっているわけではなく、執筆者によって細かく構造は変わる。例えば、研究背景と研究目的を1つにして、序論という節にまとめたり、同様に考察と結論をまとめることがある。逆に、提案手法であれば、段階的に説明するために複数の節に分割することもあれば、実験の内容を実験手法と実験結果の2つの節にわけられることもある。そのため、先頭からの順番に基づいた指定などで、必要な節を取捨選択するのは困難であると思われる。

5.2.2 要旨の構造

要旨とは、論文における要約であり、どのような研究でどのような結果が出たのかについて書かれており、他の研究者が要旨を読んで、研究の内容について凡そ把握できる必要があるものである。5.2.1 節で述べたように、要旨についても記述内容の指示があり、多くの場合は文字数の指定がある。英語の場合は凡そ 200 語から 300 語程度とされている。

要旨に含まれる内容も多くの論文で共通しており、研究背景、研究目的、手法、結果、結論が含まれている必要がある。これらを 1 つのセクションとして書く必要がある。ただし、場合によっては見出しつきで書く場合もあり、次のような見出しを使う。

- 導入
- 目的
- 研究手法
- 結果
- 結論

だが、見出し形式でも通常の場合と同様の内容を含んでいるため、形式が異なることが要旨の内容自体に大きな影響を及ぼすことはないと思われる。

各要素がどの程度、要旨に含まれているべきかというのは明確な基準は存在しない。現在、様々な論文執筆のための参考書というのは数多くあるが、参考書でも共通した基準はない。例えば、次のような配分で書くのが好ましいとする場合がある [46]。

- 総合的背景: 1 文
- 具体的背景: 1 文
- 研究テーマ: 1 文
- 研究手法: 1 文
- 結果: 1 文
- 結論: 1 文

他にも、結論の次に課題を含めるのが良いとしているものもあり [47]、要旨の構造は多種多様である。そのため、基本的に含まれている内容は共通しているが、その内容がどれほど含まれているのかは共通していないと思われる。しかし、上記で示されている内容を見ると、導入と目的であれば序論の節、研究テーマ、手法であれば手法の節と、論文の各節と要旨の内容には対応関係が存在していることがわかる。このことから、要旨に含まれるであろう節のみから、要約を生成することが有効である可能性がある。そこで、まず、序論と結論の節のみから要約を生成することが有効なのかどうかを検証する。

5.3 序論と結論からの要約生成による性能改善

先述した通り, 5.4.1 節における論文内の節と要旨の関係の仮定から, 「序論」の節及び「結論」節をキーワードにより抽出し, これらの節を用いて要約生成を行い有効性を検証する. これにより, 全文入力をして要約を生成する場合と比較して, 冗長性の少ない要約を生成することが期待される.

この手法では以下の手順で要約を実行する.

1. 要約対象となる論文より「序論」と「結論」に該当する節を抽出する.
2. 抽出した各節を結合し, 1つの文書とする.
3. 結合後に, トピックベースの抽出型要約を実施する.

最初の手順である, 要約対象の論文からの序論・結論の抽出は, 事前に設定したキーワードと一致する節名が付与されている節を対象として行う. キーワードは 5.4.1 にて示された上位 20 個に含まれる, 序論及び, 結論の節として用いられているであろう節のタイトルを使用した. キーワードの一覧を以下に示す.

「序論」に関するキーワード

'まえがき', 'はじめに', '背景と目的', '緒論', '背景', '序論', '研究背景'

「結論」に関するキーワード

'あとがき', '結論', 'おわりに', 'むすび', 'まとめ', '実験結果及び考察', 'まとめと今後の課題', 'むすび~個人と共同体の間', '授業改善への考察', 'まとめと今後の方針', '結論と今後の課題', 'まとめと今後の研究', '結び', 'まとめ, 及び今後の課題', '結論および今後の課題', '終わりに'

上記キーワードに一致する節を論文から抽出した後, 結合する. この手順を図 3.1 に示す. 抽出の際には節の名前は抽出後には含めず, また, 結合時はそのまま 1つの文書となるように, 「序論」の最後の文の後に「結論」の最初の文を結合する.

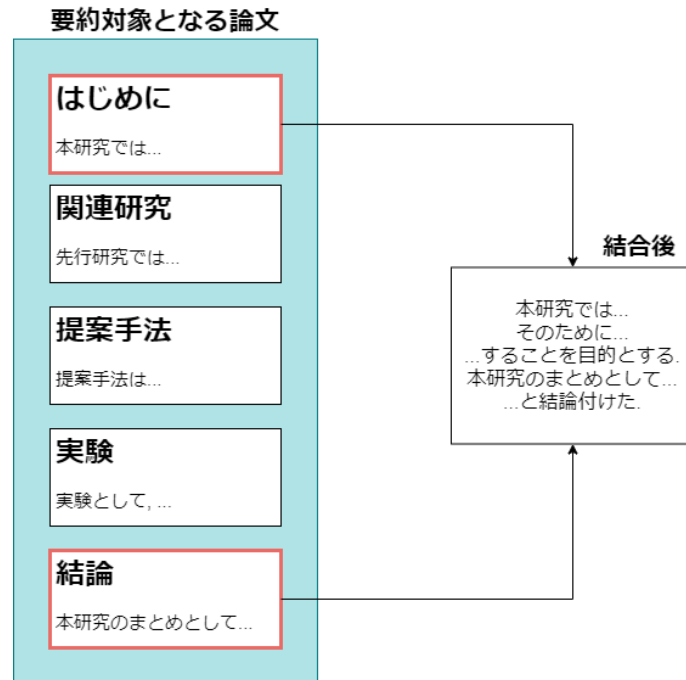


図 3.1 提案手法である論文からの序論と結論の結合の概略図. 抽出型要約を行う前に, 序論と結論に該当するセクション (赤字で囲われた箇所) を抽出して, それらを結合する. なお, 結合時にはセクション名は削除する.

これにより, 要約を行う前に必要な節を選択することが可能となると同時に, 論文要約に不要な節を省くことが可能となる. さらに, 文書内の文字数が減少するため, 要約生成における計算量の減少にもつながる.

5.3.1 実験

自然言語処理学会論文誌コーパスに収録されている 429 件の論文を対象に, 提案手法の有効性を検証した. 有効性検証として, 第 4 章にて日本語でも性能が発揮されることが示された, 自然言語処理学会論文誌コーパスに含まれるデータ数が非常に少ないため, データ数が少なくとも機能するトピックベースの抽出型要約にて, LSA によるトピックベースの抽出型要約を用いて, 全文入力による要約生成よりも改善できるか比較した. さらに LSA を LDA に置換した場合においても同様の比較を行い, 手法に依存せずに改善を行えるかを比較した.

比較には ROUGE-L, BERTScore, QAEval による評価を使用した. BERTScore を用いた理由として, BERTScore では ROUGE-L と異なり, 単語の一致ではなく, 文脈を考慮した意味的類似度による評価を行う. 抽象型要約手法の場合, 本文からの抽出ではなく, 文章の生成を行うため, 言い換えなどで正確に評価できない場合がある. BERTScore を用いることで, 論文の研究分野の内容に沿った要約を生成できているかどうかを評価した. さらに QAEval を用いて, 要旨の

内容を生成した要約の中に含ませることができているのかどうかを評価した。QAEval では、要旨内の名詞句から質問応答を生成する。そのため、生成された質問に回答するには、要旨内が持つ名詞句、もしくはそれに類似した単語を持っている必要がある。そのため、QAEval によって、要旨内の内容を持つかどうか、つまり、論文の要約を生成するのに必要な節を選択することが提案手法によって実現できているかどうかを評価した。

5.3.2 結果と考察

表 3.1 に ROUGE-L による評価結果を示す。

表 3.1 全文を要約の対象とした場合と提案手法である序論と結論に対象を限定した場合の ROUGE-L による評価の比較。

	全文	序論と結論 (LSA)
ROUGE-L	0.160	0.235
BERTScore	0.698	0.698
QAEval	0.95	1.82

また、表 3.2 に LDA の場合の結果を示す。

表 3.2 LDA における全文からの要約と提案手法による要約の評価の比較。

	全文	序論と結論 (LDA)
ROUGE-L	0.203	0.253
BERTScore	0.710	0.721
QAEval	1.07	2.64

全文入力による LSA による抽象型要約と比較して提案手法による要約は、ROUGE-L による評価と比較して、0.052 上がっていることが確認できた。これは LDA においても同様に LDA では 0.05 上昇していた。特に差が大きいのが QAEval で、LSA では 0.75、LDA では 1.57 ほど上昇しているのが確認できた。このことからわかる通り、要旨に必要な節を事前に選択し、生成に用いる自動要約手法が有効であることが示された。これは「要旨と論文内の節は対応関係にある」という仮定が正しいことを示しており、序論と結論の節に要旨に必要な内容が多く含まれていたと考えられる。また、このことから序論と結論以外の、要旨に含まれるであろう他の節を要約生成に用いることができれば、より改善を行うことが可能になるとと思われる。

5.4 節選択モデルによる論文要約手法

5.4.1 節のタイトルと要旨の構造の関係

実験より、論文における各節と要旨には対応関係にある可能性が確認できた。このことから、論文の各節は要旨の各文に対応しているという仮定が立てられる。例えば、要旨内において「本研究では を目的として、～を提案する。」という文が含まれていれば、それは論文内における序論の節の要約であると判断でき、「そして、実験を行い、以前の手法より本研究で提案した手法の方が良い結果を残した。」という文が要旨内にあれば、実験の節の要約であると判断することができる。つまり、要旨内の各文を、論文本体の各節のいずれかの要約として据えることができる。そして、節のタイトルは、節の内容を端的に表した見出しであるため、節のタイトルから序論、手法、実験、結論のいずれに該当するのか判断することができると思われる。

そこで、自然言語処理学会論文誌コーパスに収録されている論文 492 件を対象に、どのようなタイトルが論文において使われやすいのか調査した。次に使用回数の多いタイトル上位 20 個を示す。

使用されやすい節の題名 (一部)

’はじめに’，’おわりに’，’まとめ’，’まえがき’，’結論’，’関連研究’，’まとめと今後の課題’，’むすび’，’評価実験’，’実験’，’提案手法’，’考察’，’実験と考察’，’あとがき’，’評価’，’結び’，’全体的考察’，’アノテーションの不一致を引き起こす要因と対策’，’序論’，’正解データの作成’，’実験と評価’，’パターン制御指令と言語的要因との関係についての検討’，’感情主の人称制約と文のモード’，’要約手法’，’実験結果’，’自動要約への影響’，’感情表出文はなぜ一語文か’，’まとめと今後の方針’，’帰宅行動要因の分析’，’システムの評価実験’，’関連用語収集システム’，’主語割当方略および平行機能方略’，’結果’，’実験結果と考察’，’実験結果及び考察’，’韻律的特徴量の有効性’，’背景’，’研究背景’，’名詞句の指示性と数の分類’，’データ’，’決定リストを用いた局所解析’，’ILPによる多義語の曖昧性解消’，’研究概要’，’辞書からの上位語情報抽出’，’AND 検索の問題点’，’情報抽出器作成までの手順’，’動詞と主体の属性と接続関係の关系的意味’，’SkE での推量副詞とモダリティ遠隔共起の抽出’，’東日本大震災における健康危機と自然言語処理’，’機械学習による誤用タイプ分類実験’，’「伝達場面の構造」モデル’，’第1位のクラスについてのクラス所属確率推定’，’感情表現のモデル作成’，’アンサンブルクラスタリング’，’構文木付きコーパスの検索手法’，’対応付け手法の適用と要約操作’，’概念辞書の類義語と分散表現を利用した all-words WSD’，’がん用語集合の特性評価’，’形態素属性の推定’，’「です・ます」諸用法発現のメカニズム’，’補完実験’，’検索実験’，’翻訳実験’，’ダイジェストシステム’，’FrameNet における複数の汎化手法’，’授業改善への考察’，’テキストセグメンテーションに使用する表層的手がかり’，’評判情報’，’総索引と品詞タグ付きコーパス’，’要約における文再構成操作’，’単語の長さの分布に基づいた文章の分類’，’単語単位への分割’，’点予測を用いた形態素解析’，’ブートストラップ法を用いた語彙獲得における課題’，’自由回答とは’，’評価データの作成’，’構文解析処理の導入’，’関係性判定における文脈的素性の利用’，’認知単位の知覚実験’，’大域的要約と局所的要約’

使用回数の多い「はじめに」や「おわりに」といった名前からは、序論に該当する節であることや、結論に該当する節であることが容易に判断できるかと思われる。他にも「まえがき」、「あとがき」、「むすび」のように、序論と結論の節に関するタイトルは種類が少なく、キーワード一致によって容易に判定することができると考えられる。対して、手法と思しき節のタイトルや実験に関する節も含まれてはいるが、これらの多くは執筆者によって異なるタイトルをつけられていることもわかる。これらのことから、序論と結論の節との位置関係から、どの節が手法や実験にあたるのかどうか推測することが可能であると思われる。そこで、節のタイトル名一覧から要約生成に必要な節を選択する節選択モデルと、論文の各節は要旨の各文に対応しているという仮定か

ら、節ごとに要約の一部を生成する要約モデルからなる論文の自動要約手法を提案する。5.3.1 節にて序論と結論に限定した抽出型要約手法を提案し、全文を用いる要約の場合と比較して、有効であることを示した。この結果から、序論と結論以外の、手法や実験に関する節について、要約の対象として選択することができれば、より自動要約における性能を向上させることが可能になると思われる。5.2.2 において述べられている通り、序論と結論以外の内容も要旨に含まれており、より冗長性の少ない要約を作成するには他の節を選択し、要約として用いる必要がある。しかし、5.4.1 節にて述べた通り、序論や結論の節以外の節については著者ごとにタイトルを設定することが多いため、キーワード一致による選択を行うことが難しく、加えてどの節をどの程度要旨に含めるのかも著者ごとに異なる。また、抽出型要約の場合、要約を生成する際は本文から直接抽出するため、1 節あたりの文が長い場合には節の内容をまとめた要約を生成するのが難しい。なので、本研究では提案手法として、要約に必要な節をタイトル名を元に自動選択し、自動要約を行う手法を提案する。

5.4.2 提案手法の概要

提案手法は以下の 2 つのモデルから構成される。

- 入力を節、出力をその節に対応する要旨の文としてモデルを学習
- 学習時の要旨と節の対応関係の傾向を学習したモデルで、要約の生成時に使用する節を自動選択

前者のモデルを要約モデル、後者のモデルを節選択モデルと呼ぶことにする。これらのモデルは学習時はそれぞれ別で学習を行い、生成時にはこれらのモデルを併用する手法を本研究では提案する。要約時におけるモデルの全体図を図 4.1 に、生成までの流れをアルゴリズムに示す。提案手法では、要約を生成する前に節選択モデルにて、論文内の節のタイトル名から、要約生成に使用する節の番号を出力する。この番号は先頭の節を 0 番目としている。そして、節選択モデルから得られた要約生成に用いる節番号から、要約モデルへ入力する節を選択し、選択された節を要約モデルへ入力する。そして要約モデルから、それぞれの節の要約を生成し、それらを結合して論文本体の要約とする。これが提案手法における要約生成時における流れである。

要約に使用する節を事前に選択することの有効性は、5.3 節において検証した通りであるが、節選択モデルでは、論文ごとに要約に使用する節を選択するため、柔軟性の高い節選択を行うことが可能となる。加えて、節ごとに要約を要約モデルより生成することで、論文本体を全文入力して要約を生成する場合よりも、計算量を減らすことが可能となる。また、これにより、生成時の計算量を抑えると同時に、5.2.2 節で述べた、論文と要旨の関係に沿った要約生成が可能となる。

要約モデルについては、5.3 節にて用いたトピックベースの抽出型要約ではなく、ファインチューニングによる改善が確認できた mT5 モデルを用いた抽象型要約を使用した。使用した理

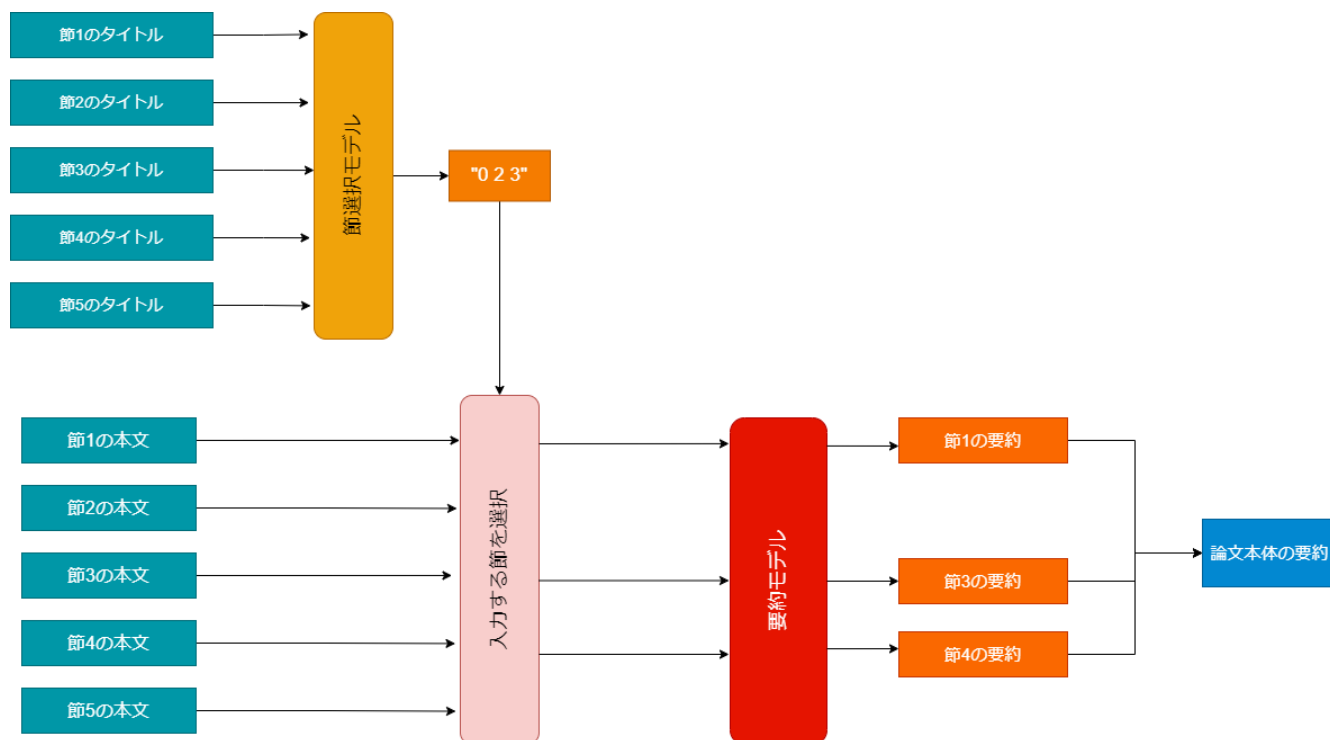


図 4.1 提案手法による論文の要約生成過程の全体図.

Algorithm 4: 提案手法のアルゴリズム. 節選択モデルを用いて要約に用いる節の番号を取得し, その番号に対応する節ごとに要約を行い, それらを結合して論文の要約とする.

Input : S_{n_1}, \dots, S_{n_N} = 論文内の各節のタイトル

St_1, \dots, St_N = 論文内の各節の本文

Output: $summary$ = 論文の要約

$t_1, \dots, t_M \leftarrow Model_{sec}(S_{n_1}, \dots, S_{n_N})$ such $0 \leq t_1, \dots, t_M \leq N - 1$; // 節選択モデルで要約対象の節番号を取得

for $i \leftarrow 1, \dots, M$ **do**

$secsum[i] \leftarrow Model_{sum}(St_{t_{i+1}})$; // 対象のセクションごとに要約を生成

end

$summary \leftarrow Concat(secsum)$; // 得られた節ごとの要約を結合

由として、抽象型要約では文章をゼロから生成するため、複数の文の内容をひとつにまとめた文を生成させることが可能となるためである。抽出型要約の場合、文単位で抽出するため、複数の文を1つの文にまとめることができないため、冗長性が増してしまう。

次に、本研究の提案手法の要約モデルと節選択モデルについて解説する。

5.4.3 要約モデル

提案手法における要約モデルでは、通常の seq2seq モデルと同様の方法で学習を行う。異なる点として、入力は論文本体ではなく、論文の各節であり、出力はそれに対応する要旨本文内の文である。そのため、学習時には要旨内の各文と論文内のいずれかの節を対応付けさせる必要がある。そこで、本研究では 5.2.2 節で述べたように、要旨は論文におけるいずれかの節の要約であるという仮定から、DANCER[7] と呼ばれる要約モデルの学習法を用いてモデルの学習を行う。DANCER では以下の手順で、要旨内の各文と論文内の節の対応付けを行う。

1. 要旨内のある文 s_i について論文全文それぞれと、ROUGE-L スコアによるスコアリングを行う。
2. スコアリング後、もっともスコアの高い本文 c_j が属する節 S_k を探索
3. 節 S_k を、文 s_i に入出力関係として対応付けする。

この対応付けを行った学習データを構築し、要約モデルの学習を行う。

DANCER を使用する最大の利点は、学習時におけるデータ数の増加が見込めることである。例えば、自然言語処理学会論文誌コーパスに上記手順を適応して学習データの構築を行った場合、表 4.1 に示すように変化する。DANCER によって大きく増加し、1050 件に増えることで大型の

表 4.1 DANCER を適用する前と後のデータ件数。凡そ 3 倍ほどに増加している。

	元データ	DANCER 適用後
学習データ件数	343	1050

事前学習モデルであれば、十分に学習が可能なデータ件数を確保することが可能となる。最後に要約モデルにおける生成過程の全体図を図 4.2 に示す。

5.4.4 節選択モデル

節選択モデルでは、先述の通り、要旨内の文章と節には対応関係があるとして、その対応関係の傾向を学習し、生成時に要約モデルに入力する節の選択を行うモデルである。モデル学習のための学習データは、5.4.3 にて構築した学習データを利用し、節のタイトル一覧を入力として、要旨の文を対応付けされた節の番号を出力とするデータを構築する。次のような構造となるように

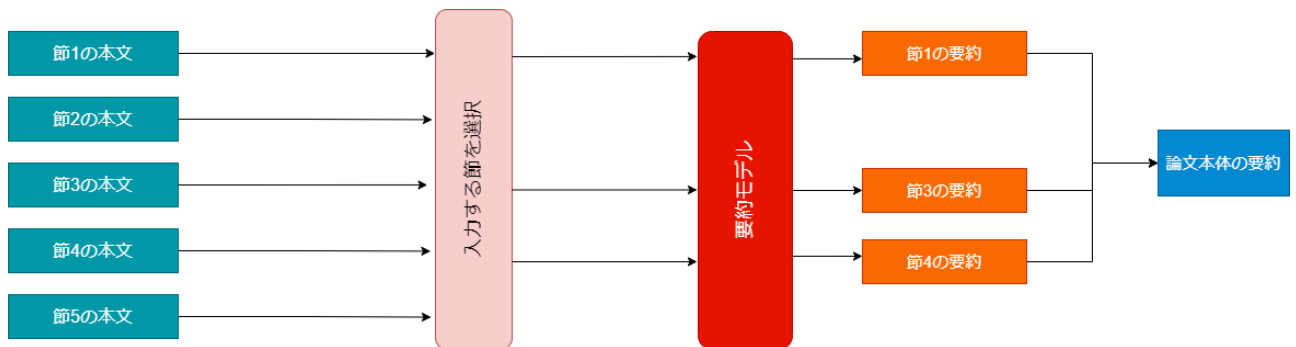


図 4.2 要約モデルにおける要約の生成過程の図.

データを構築する.

学習データの構造例

入力 "序論 手法 実験 考察 結論"

出力 "0, 1, 4"

つまり, 入力として論文内の各節の題名を半角空白で結合し, DANCER を用いた際に要旨に割り当てられた節の先頭からの番号 (開始は 0) を出力とする学習データを構築する.

節の題名单独で入力するのではなく, 全てを連結し, 入力とすることで, 前後関係を考慮させた学習を行わせることが期待できる. 例えば, 次のような節の題名を設定している論文があったとする.

1. はじめに
2. 関連研究
3. 提案手法
4. 実験
5. 実験結果
6. 結論

このとき, 「はじめに」や「提案手法」, 「結論」というのは前後関係がなくとも, 研究における重要な内容を含んでいる節であると判断できる. 対して, 「実験」の場合, 初めて後ろに「考察」の節があって, 結果も含めて実験についての重要内容を書いているかもしれないと判断可能である. 対して, 今回の場合では直後に「実験結果」の節が来ているため, 実験に対する考察が「実験結果」にも含まれていて, より重要である可能性が高いと判断できる.

また, 節選択モデルでは多言語モデルを利用した, few-shot cross-lingual transfer を行う. Cross-lingual Transfer とは学習するデータの言語と解くタスクの言語が異なる際に行う転移学習の一つである. few-shot とは, 事前に学習したモデルに少量学習データを与えたうえで, その

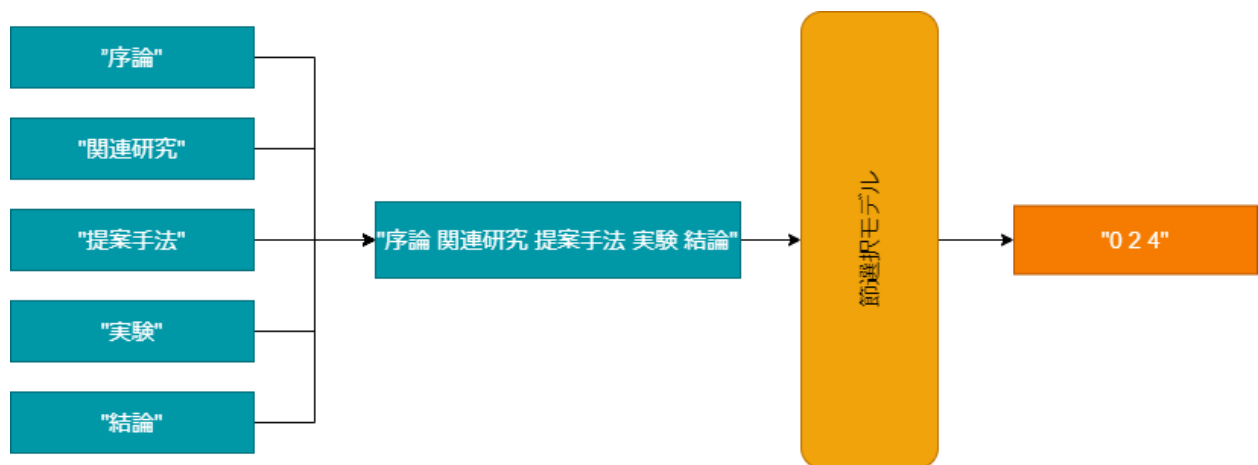


図 4.3 節選択モデルにおける生成の流れ. 図内部におけるセクションは節のことを指す. 図の例では, 「序論」「関連研究」「提案手法」「実験」「結論」の 5 つの節から成り立っている. これらを半角スペースで結合して一つの文として節選択モデルに入力して, "0 2 4" という出力を得る.

学習データに対するタスクを解くことを指す. つまり, 学習データの多い言語で学習を行った後に, 学習データの少ない言語で追加で学習するのが few-shot cross-lingual transfer である. 多言語モデルを用いて, コーパスの数が多い英語で学習し, 学習に使用したコーパスと出力形式が同一であるタスクで性能が改善されることが確認されている [48].

そこで本研究では, 既に英語と日本語共に学習データがあることを利用し, 英語のコーパスによる学習を行った後, 日本語コーパスによる few-shot cross-lingual transfer 学習を行う. 英語の学習データとして, Discourse-Aware Attention Model の学習に使用された, 論文要約コーパスである arxiv コーパスを使用する. 学習データとして公開されている 21 万 5 千件の論文のうち, 94% を使用する. 節選択モデル用の学習データへの構築手順は先述の通りである.

また, 入出力について, テキスト形式で行うように学習を行わせる. 論文において節の数というのは著作者によって大きく異なる. 同様に, 要旨に用いられる節の数も異なってくる. そのため, リストなどの固定長形式では対応しきれない. そこで, テキスト形式で入出力を行うことで可変長として対応することが可能となる. 可変長となるため, 著作者ごとに節の数も異なっても, 要旨に対応する節の数も変化しても, 文字数の違いとして処理することが可能となる.

最後に節選択モデルにおける節の選択の流れを図 4.3 に示す.

5.4.5 全文からの要約生成との比較

提案手法が, 全文を用いた抽象型要約よりも有効なのか, どうか比較した. さらに, 節選択を行うことが有効なのかどうかも確認するために, 全文からの生成に加えて, 節選択を行わず全節を

要約モデルに入力し、提案手法と同様の手順で生成した場合とも比較した。

提案手法における要約モデルは、日本語においてもファインチューニングによって性能を発揮することが確認された多言語事前学習モデルである mT5 を使用した。節選択モデルも同様に mT5 を使用した。要約モデルの学習データには自然言語処理学会論文誌コーパスに収録されている日本語で記述されている 429 件の論文のうち、343 件を使用した。節選択モデルの学習には、前述の通り、arxiv.org コーパス内の論文 21 万 5 千件のうち 94% と、要約モデル構築に使用した自然言語処理学会論文誌コーパス内の論文 343 件を使用した。節選択モデルと要約モデルのそれぞれで学習時における設定は、以下に示す。

arXiv コーパスによる節選択モデル学習時の設定

学習データ数 約 21 万 5 千
バッチ数 16
エポック数 5
入力可能最大長 100 トークン
出力可能最大長 50 トークン

自然言語処理学会論文誌コーパスによる節選択モデル学習時の設定

学習データ数 343
バッチ数 8
エポック数 10
入力可能最大長 100 トークン
出力可能最大長 50 トークン

要約モデル構築時の設定

学習データ数 1050
バッチ数 4
エポック数 10
入力可能最大長 500 トークン
出力可能最大長 100 トークン

比較対象となる全文入力による要約モデルは、提案手法の要約モデルと同様に mT5 を使用した。構築時の学習データは提案手法と同様に自然言語処理学会論文誌コーパス内の 343 件を使用した。比較対象の全文入力要約モデルは以下の設定で構築した。

全文入力モデル構築時のパラメータ

学習データ数 343
バッチ数 4
エポック数 10
入力可能最大長 1000 トークン
出力可能最大長 500 トークン

評価には提案手法の要約モデル構築にて使用しなかった、自然言語処理学会論文誌コーパス内の残り 86 件を使用した。評価手法には ROUGE-L と BERTScore, そして QAEval を使用した。また、有意差検定には Dunn の検定を用いた。

5.4.6 実験結果と考察

提案手法, 及び比較対象の評価の結果を表 4.2 に示す。

表 4.2 全文入力によって要約を生成した場合と節選択モデルを用いて要約を生成した場合の各評価の結果。全文入力からの生成, 及び全節入力からの生成に対して有意水準 $p < 0.01$ における有意差を示した場合, それぞれ *, ** として示している。

評価手法	全文入力	全節入力	節選択モデル
ROUGE-L	0.223	0.256	0.274*
BERTScore	0.623	0.723	0.725*
QAEval	3.65	2.35	5.49*,***

ROUGE-L, BERTScore, QAEval, それぞれの評価において, 節選択モデルを用いた要約生成のほうが, 全文入力による要約生成よりも高い結果となった。このことから提案手法は全文入力よりも要約性能は高く, さらに序論と結論のみの要約と比較して要約性能が高いことが確認できた。序論と結論のみに限定した抽出型要約と比較して, 節選択モデルによる抽象型要約のほうが QAEval における評価が高い。これは mT5 を要約モデルに使用したことも要因として考えられるが, 全文入力も同様に mT5 を用いていることを考慮すると, 要約生成に使用している手法の違い以上に, 事前の節選択において, 序論と結論以外の節も選択していることが有効に働いていると考えられる。

また, QAEval の評価において序論と結論の抽出型要約よりも, 全文からの抽象型要約のほうが高い結果となった。抽出型要約では文を直接抽出して要約とするため, 複数の文をまとめることができず, 解答に必要な語句を多く含めることができなかつた。対して, 抽象型要約の場合は, 言い換えなどによって複数の文を 1 つの文にまとめるように生成することが可能である。そのため, 全文を用いた抽象型要約と比較して序論と結論を用いた抽出型要約の評価が低いことにつな

がったのではないかとと思われる。しかし、同じ抽象型要約である提案手法では QAEval の評価が高いことから、要旨に含むべき内容を持つ節を提案手法では選択できていると思われる。同様に全節入力における QAEval の値が、全文入力より低いのは、QAEval に使用する質問生成モデルと質問応答モデルにおいて、入力最大長に対する制限により、全文を用いての生成ができなかったことが要因であると思われる。そのため、全文を入力することができれば QAEval における評価は提案手法と同様となるとと思われるが、このことは、全節から生成する場合においてはむしろ冗長な要約を生成することを示していると考えられる。これは ROUGE-L における適合率においても示されており、全節から生成する場合の適合率と比較して、提案手法による生成における適合率の方が高いことが確認された。このことから、提案手法による論文要約生成は冗長性の少ない要約の生成が行えていると判断できる。

さらに、表 4.3 に各手法の ROUGE-L 及び、BERTScore の再現率と適合率を示す。ROUGE-L

表 4.3 各手法における ROUGE-L と BERTScore における詳細な結果

	全文入力	全節入力	提案手法
ROUGE-L(再現率)	0.206	0.321	0.242
ROUGE-L(適合率)	0.257	0.256	0.383
BERTScore(再現率)	0.646	0.733	0.748
BERTScore(適合率)	0.603	0.742	0.713

における結果では、再現率においては全節を用いて生成した場合よりも低いが、適合率が高い結果が確認された。全節を入力して生成した場合、全文を網羅して生成することになるため、必然的に要旨と同じ単語をより多く含むことになる。そのため、再現率においては提案手法よりもスコアが高くなったものと思われる。しかし、適合率が高いことから、冗長性は提案手法の方が少ない文章の生成ができていると考えられる。これは節選択モデルによって、事前に要約対象となる節を論文ごとに選択することで、重要な節のみを用いて要約を生成していることが要因であると思われる。また、BERTScore では再現率において、提案手法のほうがスコアが高かった。これは文章全体の意味的に、提案手法による要約生成のほうがより似ていると考えられる。これも節選択モデルで要旨に含まれている節を選択する、つまり要旨内の文章と意味的に類似した節のみを使用しているため、このような結果に繋がったと思われる。これらの結果と QAEval における結果を踏まえると、本研究の提案手法は、意味的に要旨と類似性を持たせ、冗長性を減らしながらも、要旨に含むべき情報が多い論文要約を行うことが可能である。

5.5 本章のまとめ

本章では、論文と要旨の構造の関係性を利用した要約手法を提案した。まず、序論と結論の2つの節に限定する手法について述べ、全文からの要約と、LSAによる抽出型要約を比較した。比較の結果、全文入力より性能が高いことが確認できたため、さらに序論と結論だけではなく、他の節も要約に使用する抽象型要約手法を提案した。結果、この手法は全文からの要約だけではなく、序論と結論の節のみからの要約と比較して、有効であることが確認できた。つまり、論文と要旨の構造の関係に着目し、節のタイトル名から要約に使用する節を選択して、要約生成を行う本研究の提案手法が、全文から要約を生成するよりも有効であることが示された。

第 6 章

汎用言語モデルと提案手法の性能比較

6.1 背景

本研究では提案手法として、論文の構造と要旨の構造に着目し、要約に用いる節を自動選択する抽象型要約手法を提案した。本章では、提案手法が汎用言語モデルと比較して、論文要約性能が高いのかどうかの検証について述べる。比較対象として、大型の事前学習モデルである GPT を用いた、対話型 AI ツール chatGPT[49] を用いた。chatGPT は大型事前学習モデルであり、汎用言語モデルである GPT-3 が元になっており、要約生成においても高い性能を発揮すると思われる。この chatGPT と比較を行い、提案手法が論文の要約において汎用言語モデルよりも高い性能を発揮するのかどうかを検証した。検証の際には、いくつかの論文を用いて、chatGPT より要約を生成させ、提案手法の生成する要約との評価の比較を行った。比較実験について述べる前に chatGPT について解説する。

6.2 chatGPT

chatGPT は AI 研究及び開発を行う企業である OpenAI[50] より公開されている、大型事前学習モデルである GPT を用いた対話型 AI ツールである。chatGPT ではユーザーの入力された内容に基づいて、返答する形で出力を生成する。図 2.1 に chatGPT の入出力例を示す。オレンジ色のアイコンがある箇所がユーザーの入力であり、緑色のアイコンがある箇所が chatGPT からの出力である。文章の内容から判断できる通り、chatGPT はユーザーからの質問に対して対話的に返答するという形で出力を行っている。また、内容に沿って解答しており、図 2.1 では要約の工学化について質問し、自動要約についての内容を返答している。さらに、手順について教えてほしいという質問についても対応しており、出力内容も手順となるように、番号付き箇条書きで出力をしている。

さらに chatGPT では、情報の補填を行うように指示することで、これまでの chatGPT 自身の



要約という作業の工学化は可能なのでしょうか。仮に可能だとした場合どのような手順で要約が行われるのでしょうか。



要約の工学化は可能です。実際、自然言語処理（NLP）の分野では、機械学習やディープラーニングを用いて自動要約の技術が開発されています。



自動要約は、大量の文章から重要な情報を抜き出し、短い要約文を生成するプロセスです。以下は、自動要約の手順の概要です。

1. 入力テキストの前処理：入力されたテキストを正規化し、不要な情報を取り除くことで、処理の精度を向上させます。
2. 文章分割：入力テキストを文章に分割します。
3. 単語分割と品詞タグ付け：各文を単語に分割し、それぞれの単語に品詞タグを付与します。
4. 重要度の算出：単語の重要度を算出します。重要度は、その単語がテキスト全体でどの程度出現するか、その単語が文脈上でどの程度重要ななどの要素に基づいて計算されます。
5. 要約文の生成：重要度の高い単語やフレーズを含む文を選択し、それらを組み合わせて要約文を生成します。生成された要約文は、元の文章の内容を的確に伝えるように注意深く構成されます。

自動要約技術は、大量の文書を迅速かつ正確に処理することができます。しかし、完全に自動化することは難しい場合もあります。そのため、人手で修正や編集が必要な場合もあります。

図 2.1 chatGPT における入出力例。オレンジ色のアイコンがユーザーであり、緑色のアイコンが chatGPT である。

出力内容を基に情報を創り出し補填することも可能である。図 2.2 にその様子を示す。図 2.2 では、架空の種族についての設定を与え、想像してほしい、という命令を与えると、与えられていない文化の内容について chatGPT 側が詳細に述べている。さらに、出力内容にでてきた祭りの内容に質問すると、その内容についても掘り下げて解答しているのが確認できる。

このように chatGPT は文脈を考慮した出力を行うことが可能である。chatGPT は InstructGPT[51] が基となっている。次の節から InstructGPT について解説する。


6.3 InstructGPT

InstructGPT は大型事前学習モデルである GPT の発展形である GPT-3 に対して、人間の好みに合うように出力を行うよう調整されたモデルである。GPT-3 は第 2 章において述べた通り、Transformer の Decoder のみで構成されており、基本となる GPT と異なる点はパラメータ数

HS バンブーエルフは以下のような特徴を持ちます。

- 主食は竹であり、青竹をかみ砕く強靱なあごをもつ。
- 竹を用いた鎧や服装、武器を作る。
- 主に竹林に住んでいる。
- パンダを騎乗動物として飼っており、パンダライダーと呼ばれる騎兵が存在する。
- パンブードラゴンと呼ばれる守護神がいる。


これらを元にバンブーエルフの普段の生活について、想像してみてください。

 この設定に基づいて、バンブーエルフは竹林に住んでおり、竹を主食としています。彼らは竹を用いて鎧や服装、武器を作り、パンダを騎乗動物として飼っています。また、パンブードラゴンと呼ばれる守護神を信仰しています。

✖

バンブーエルフの生活は、竹の森での採集や鎧作り、パンダの育て、竹に囲まれた家族や村の生活などが主要要素です。彼らはパンダライダーと呼ばれる騎兵として、竹林を守ったり、他の町や村を訪問したりすることもあります。また、パンブードラゴンを信仰しているため、祭りや祈りの儀式が重要な部分であり、彼らの文化や歴史に根深い影響を与えています。

HS それではバンブーエルフが執り行っている祭りについて想像してみてください。

 この設定に基づいて、バンブーエルフはパンブードラゴンを信仰しているため、彼らは定期的にパンブードラゴンを祭る祭りを開催します。この祭りは、村や町全体が参加する大規模なものから、家族や小さなグループで行われるものまで様々です。

✖

祭りの主な内容は、祈りや歌、踊り、食事などです。また、竹を用いた装飾や彫刻なども重要な要素となります。彼らは、このような祭りを通じてパンブードラゴンへの敬意や祈り、また村や家族の結束を強も目的としています。また、この祭りを通じてバンブーエルフの文化や伝統が伝承されています。

Regenerate

図 2.2 chatGPT での情報補填の様子。存在しない架空の種族の設定を数例与えることで、生活や文化の内容について詳細に述べると同時に、自身の出力内容に対する質問にも解答ができている。

と学習データ数である。第 2 章で述べた GPT の事前学習方法のみで、特定のタスクにファインチューニングせずとも解くことができる汎用性が高いモデルであるが、出力内容が不正確、もしくは道徳的に好ましくないと言われる出力がされることもあるという問題点がある。この問題点を解消するために、InstructGPT では人間の評価をフィードバックする強化学習を行うという手法をとる。

InstructGPT では以下の手順で学習を行う。

1. 教師ありのファインチューニングを行い、モデル (SFT モデル) を構築する。

2. 人手による評価を模倣する報酬モデルを構築する.
3. 報酬モデルからの評価が最大になるように SFT モデルを再学習させる.

各手順について、解説する.

6.3.1 SFT モデル

SFT モデルとは、元となる GPT-3 に対して、人手で作成したデータセットを用いてファインチューニングを行ったモデルである。学習に使用したデータセットは以下のような種類のデータから構成されている。

Plain 人手作成した、任意のタスクについて記述されたプロンプト

Few-shot プロンプト内にタスクについての解答例を記述したうえで、タスクを解くように命令するプロンプト

User-based OpenAI の API を使用するアプリケーションのユースケースに対応したプロンプト

これらのデータセットは SFT モデルの構築だけでなく、報酬モデルの構築や、報酬モデル構築後の再学習にも使用される。

6.3.2 報酬モデル

報酬モデルは SFT モデルの出力に対する人手の評価を模倣するように学習するモデルである。人手の評価は、SFT モデルの学習に使用されたデータセットを SFT モデルに入力し、SFT モデルから得られる出力に対して行われる。出力は 1 つの入力に対して 4 から 9 個を得て、人手によって得られた出力に対してランク付けを行う。この人手のランク付けを報酬モデルは学習する。

報酬モデルは、SFT モデルと同様に GPT-3 を用いているが、異なる点として、GPT-3 の最終出力を単語ではなく、スカラー値になるように出力層を取り替えたものになっている。このスカラー値で SFT モデルからの出力についての評価を行う。この評価の流れを図に示す。そのため、報酬モデルの学習では評価値を使って損失関数を計算する。SFT モデルへの入力を x として、SFT モデルからの出力を y とする。このとき、ランクが高い方の出力を y_w 、低い方を y_l とする。このとき、報酬モデルからの出力を $r_\theta(x, y)$ と表すとした場合、人手による評価データセット D において、 $r_\theta(x, y_w) - r_\theta(x, y_l)$ が最大化すればよい。さらに、SFT モデルから得る出力は複数あるため、出力の数を K としたとき、得られる出力の組み合わせは $\binom{K}{2}$ 組となるため、最終的には全ての組み合わせに対する、 $r_\theta(x, y_w) - r_\theta(x, y_l)$ の期待値を最大化すればよいので、報酬モデル

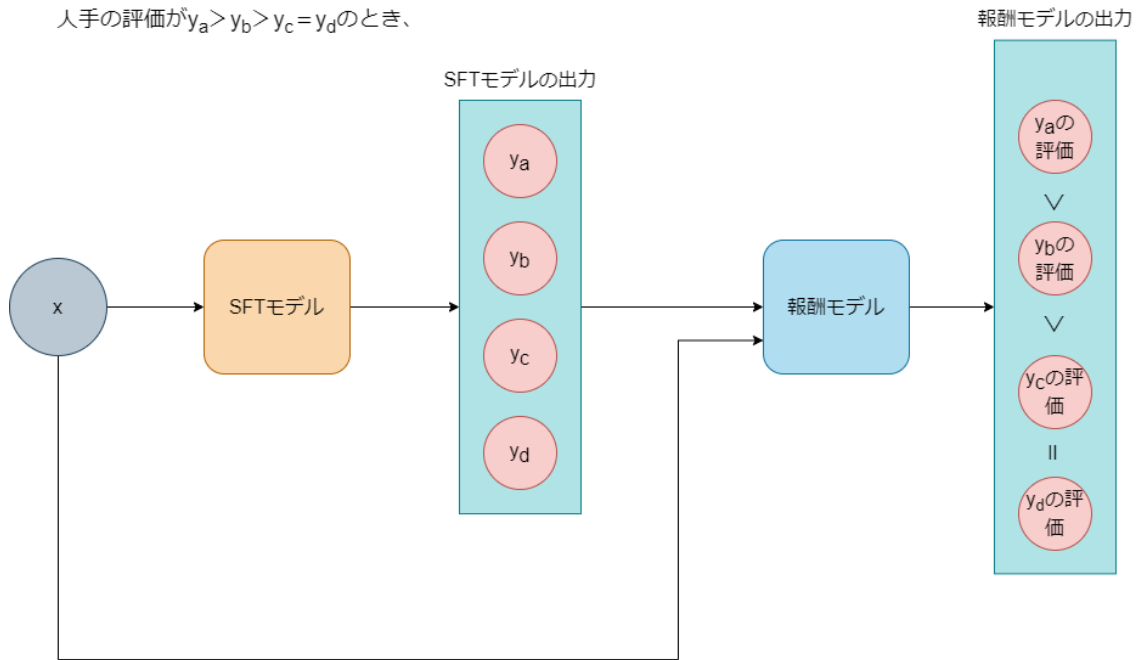


図 3.1 報酬モデルの出力までの流れ. 入力 x に対する SFT モデルの出力が y_a, y_b, y_c, y_d のとき, 人手の評価が $y_a > y_b > y_c = y_d$ であるとする. この時報酬モデルも同様の出力を行うように学習する.

に対する損失関数 $loss(\theta)$ は次の式で表される.

$$loss(\theta) = -\frac{1}{\binom{K}{2}} E_{(x, y_w, y_l \sim D)} [\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))] \quad (3.1)$$

この損失関数を用いて報酬モデルを学習していく.

6.3.3 SFT モデルの再学習

報酬モデルを構築したのちは, 報酬モデルの出力 $r_\theta(x, y)$ が最大になるように SFT モデルを再学習する. ただ, 報酬モデルの出力を最大化するようにそのまま学習をさせた場合, 報酬モデルが最大になるような出力を行うようになり, 実際出力内容が元の SFT モデルから大きくかけ離れてしまう可能性がある. そこでペナルティとして, KL ダイバージェンスによるペナルティを設ける.

これはある 2 つの確率分布に対する差異の指標であり, ある 2 つの確率分布 P, Q における, 事象 x が発生する確率を $P(x), Q(x)$ としたとき, 確率分布 P に対する期待値 $E_p \left[\log \frac{P(x)}{Q(x)} \right]$ として表現される. GPT-3 は Transformer の Decoder から構成されるため, 最終出力は入力 x に対する出力 y についての語彙確率分布 $\pi(y|x)$ である. なので, 再学習を行う前の SFT モデルからの語彙確率分布を $\pi^{SFT}(y|x)$, 再学習する SFT モデルの語彙確率分布を $\pi_\phi^{RL}(y|x)$ とすると,

KL ダイバージェンスによるペナルティを用いた再学習時の損失関数 $objective(\phi)$ は,

$$objective(\phi) = E_{(x,y) \sim D_{\pi_{\phi}^{RL}}} \left[r_{\theta}(x,y) - \beta \log \frac{\pi_{\phi}^{RL}(y|x)}{\pi^{SFT}(y|x)} \right] \quad (3.2)$$

となる. β はペナルティ項に対する係数である. さらに, 元の GPT-3 の汎化性能を損なわないように GPT-3 に用いたデータセットを使って, 再学習モデルからの出力 $\pi_{\phi}^{RL}(x)$ に対する対数尤度をペナルティ項として追加する. つまり, 最終的な損失関数は

$$objective(\phi) = E_{(x,y) \sim D_{\pi_{\phi}^{RL}}} \left[r_{\theta}(x,y) - \beta \log \frac{\pi_{\phi}^{RL}(y|x)}{\pi^{SFT}(y|x)} \right] + \gamma E_{x \sim D_{pretrained}} [\pi_{\phi}^{RL}(x)] \quad (3.3)$$

ここで γ は対数尤度によるペナルティ項に対する係数である.

InstructGPT では, SFT モデルの再学習を終えた後は, 再学習した SFT モデルで報酬モデルを再学習し, という繰り返しを行う. これを Reinforce Learning from Human Feedback と呼び, 人間が尤もらしいと感じる出力を行えるようになる.

6.3.4 InstructGPT と chatGPT の違い

chatGPT は InstructGPT と学習方法は変わらないが, 異なる点は SFT モデルの学習で使用するデータセットである. InstructGPT では, タスクを解くための命令が書かれたプロンプトをデータセットとして使用したが, chatGPT では対話方式からなるプロンプトを用いて学習を行っている. さらに報酬モデルの構築には対話内容についてランク付けを行ったものを使用している.

6.4 chatGPT との論文要約性能の比較

6.4.1 概要

背景にて述べたように本章では chatGPT と提案手法の論文要約性能に対して行った比較実験について述べる. 評価には自然言語処理学会論文誌コーパス内の, 提案手法の要約モデル構築に使用しなかった 86 件の中から, ランダムに 4 件を使用した. 提案手法における要約の生成は, 第 5 章にて述べた手法と同様の方法で行った. 対して, chatGPT は対話方式によってタスクを解くため, 以下の手順で要約生成を行った.

1. 最初の入力として, 「今から複数の文を入力します. 記憶してってください。」として, chatGPT にこれらかの入力を記憶するよう命令する.
2. 節ごとに入力を行う.

- すべての節を入力し終わったら、「これで終了です。さて、ここまで入力された内容について要約してください。」と入力して、要約を生成させる。

この手順を、第5章にて評価用データとして使用した、自然言語処理学会論文誌コーパスから評価対象の論文4件に対して行い、生成された要約を chatGPT に対する評価に使用した。

評価手法として、ROUGE-L と BERTScore, そして QAEval を使用した。これまでの実験同様に、ROUGE-L による単語の一致、意味的類似度、情報の被覆度を評価することで、chatGPT と提案手法のどちらが、より要旨の内容とほぼ類似した要約を生成できているかどうかを比較した。さらに、要約モデルを chatGPT に置換した場合とも比較し、本研究の論文要約における有効性を検証した。

6.4.2 結果と考察

chatGPT と提案手法それぞれの評価結果を表 4.1 に示す。結果より、ROUGE-L, BERTScore,

表 4.1 chatGPT と提案手法の要約性能の評価結果

	chatGPT	提案手法	提案手法 +chatGPT
ROUGE-L	0.262	0.332	0.289
BERTScore	0.722	0.742	0.747
QAEval	1.61	1.76	2.40

QAEval のいずれにおいても提案手法における評価が高いことがわかった。このことから、本研究の提案手法は chatGPT による要約性能よりも高いことが示された。加えて、QAEval においては提案手法と chatGPT を併用した場合が最も高く、本研究の提案手法が汎用言語モデルよりも性能が高いことだけでなく、汎用言語モデルを論文要約に特化させることが可能であることが示された。

しかし、ROUGE-L におけるスコアの結果では、chatGPT と併用しない場合が最も高い。chatGPT では入力の内容を保持して要約を行うが、汎化性能が高いため、論文に特化した要約を行うことが困難であると思われる。対して、本研究の提案手法は論文に特化し、さらに要旨の生成に特化している手法になっている。例えば、「本研究では～」のような要旨特有の言い回しを含んだ要約生成を行うことが難しい。つまり、要約の対象となる文書の種類に対応した要約を行えるかどうかがこの ROUGE-L におけるスコアの差につながったのではないと思われる。このことから、自動要約においては汎化性よりも対象となる文書の種類に応じた手法を用いることが重要であるのではないと思われる。ただし、QAEval における評価が提案手法と併用した場合が最も高いことから、要旨に含まれる情報を生成する文に含めるという点においては提案手法において用いている要約モデルよりも chatGPT のほうが性能が高いということがいえる。

さらに、表に ROUGE-L と BERTScore の再現率と適合率を示す。表よりわかるとおり、提案

表 4.2 ROUGE-L と BERTScore における評価結果の詳細

	chatGPT	提案手法	提案手法 +chatGPT
ROUGE-L(再現率)	0.271	0.346	0.282
ROUGE-L(適合率)	0.260	0.339	0.311
BERTScore(再現率)	0.719	0.741	0.747
BERTScore(適合率)	0.724	0.745	0.740

手法が再現率と適合率の両方において、chatGPT よりも高い。つまり、chatGPT よりも、論文要約として適切な要約の生成を行うことができているということがわかる。また、要約モデルを chatGPT に置換した場合は、ROUGE-L における適合率、及び BERTScore の再現率・適合率の両方で chatGPT のみの場合よりも高いスコアが確認できた。これは、提案手法と併用することで、節選択モデルにて事前に論文要約に必要な節のみが chatGPT に与えられるため、より論文要約に近い要約の生成を行うことができるようになったと考えられる。逆に言えば、chatGPT 単体で論文要約を行った場合、必要な情報が消えてしまっている可能性があるということでもある。

以上の結果から、提案手法は学習データ数が少ない、もしくは論文要約に特化していない言語モデルにおいて、論文要約としての性能を改善させることが可能であることが示された。このような結果が出た要因として、提案手法の節選択モデルが挙げられる。ROUGE-L における適合率の比較を見てわかるとおり、提案手法及び、提案手法と併用した chatGPT による要約では著しく改善されていることがわかる。この結果から、節選択モデルによって事前に重要な節に要約の対象を限定することで、冗長性を減らすことにつながっていることがわかる。また、要約モデルを論文に特化していないものに置換することで、論文要約における性能が改善されたことから、本研究の提案手法において節選択モデルが論文要約における性能改善において重要な役目を果たしていると考えられる。

6.5 本章のまとめ

本章では提案手法に対して、汎用言語モデルである GPT-3 を用いた対話型 AI ツールである chatGPT と、論文の要約における性能比較を行った。chatGPT は GPT-3 の汎用性を持ちながら、より人間の好み合うような出力が行うように調整されている。論文要約における性能比較をしたところ、提案手法が ROUGE-L, BERTScore, QAEval の 3 項目において評価が高いことが判明した。このことから、提案手法が論文要約において有効な手法であることが示された。また、chatGPT と提案手法を併用することで chatGPT のみで要約した場合と比較して改善したため、提案手法に含まれる節選択モデルによる事前の節選択が、論文要約において肝要であることが判

明した.

第7章

まとめ

7.1 構成

本章は以下の構成から成る.

- 本研究のまとめ
- 提案手法に対する考察と課題
- 今後の展望

本章では, 最初に本研究のまとめを行う. 研究背景と目的, 提案手法と有効性の結果についてまとめる. 次に提案手法に対する考察をまとめる. 考察では節選択モデルの性能や, 要約生成モデルの性能について評価結果だけでなく, 出力内容についても考察を行い, 提案手法の問題について述べる. 最後に本研究の今後の展望について述べる. 提案手法の応用先や今後解決すべき課題についてまとめる.

7.2 本研究のまとめ

昨今において, 自動要約における研究は盛んであり, 日本語においても学習用のコーパスが公開されるなど, 言語を問わない. しかし, 自動要約の研究の多くはニュース記事のような短い文書を対象として行われており, 論文などの長文文書を対象とした研究は数が少なく, 特に日本語を対象とした長文文書の自動要約研究は希少である. そこで, 本研究では, 日本語における長文文書の自動要約として, 日本語で記述された論文を対象とした自動要約手法を提案し, 提案手法有効性を示すことを目的とした.

本研究では, 論文要約を行う前に, 既存の自動要約手法が日本語においても性能を発揮するかどうかを検証した. 既存の自動要約手法の多くは主に英語のコーパスを対象に行われており, 日本語における性能は不明である. そこで, 英語以外の言語でも性能を発揮すると示唆されている

潜在的意味解析による抽出型要約と多言語事前学習モデル mT5 による抽象型要約手法の性能を評価した。評価の結果、潜在的意味解析による抽出型要約では英語と同等の性能を発揮し、mT5 による抽象型要約では、日本語コーパスのみによるファインチューニングによって性能が改善したことを確認できた。本研究ではこれらの手法を用いて論文要約を行った。

過去の論文を対象とした論文の自動要約の研究の多くでは、論文の節構造に着目した手法が多く、例えば Discourse-Aware Model のような、節と本文の関係をニューラルネットワークの構造に落とし込んだモデルがある。本研究では、論文の構造だけでなく論文の要約である要旨の構造との関係に着目した手法を提案した。まず、本研究では序論と結論の内容が必ず要旨に含まれていることから、これら 2 つの節から論文の要約を生成する抽出型要約手法を提案した。結果、全文から生成する場合よりも序論と結論の節から要約を生成することが有効であることを確認できた。また、序論と結論の節から要約を生成する手法は、論文と似た構造を持つ評論文においても有効であることも確認できた。

この結果から本研究では序論と結論以外の節も用いて論文の要約を行う抽象型要約手法を提案した。提案手法は要約に使用する論文内の節を自動で選択する節選択モデルと、節の内容を要約する要約モデルの 2 つから構成される。提案手法では、節選択モデルによって選択された節それぞれから、要約モデルを用いて要約を生成し、節それぞれの要約を結合して、論文の要約とした。これにより、冗長性の少ない要約を生成できることが期待でき、さらに節ごとに要約を生成するため、計算量を抑え、さらに学習データを増やすことで、要約性能が向上することが期待された。提案手法が実際に論文の要約において有効かどうかを検証するために、全文を要約の対象とした場合と比較して高い性能を発揮することを確認し、さらに序論と結論のみを要約生成に使用する場合よりも性能が高いことを確認した。

さらに、本研究では汎用言語モデルとの論文要約性能の比較も行った。汎用言語モデルとして、OpenAI より公開されている対話型 AI ツールである chatGPT を使用し、提案手法と論文の要約における性能を比較した。chatGPT は大型事前学習モデルである GPT-3 を、人間が好ましいと判断できるような出力を行うよう調整されている。比較の結果、chatGPT よりも提案手法が性能が高いことが確認された。よって、これまでの実験の結果から、本研究の目的である、日本語論文要約の提案手法の有効性を示すことができたと判断できる。

7.3 提案手法に対する考察と課題

7.3.1 要約前の節の選択

本研究の提案手法である要約を生成する前に、論文と要旨の関係性から要約の対象を限定する手法は有効であることが実験によって示された。特に、論文ごとに要約対象となる節を選択する節選択モデルによる節選択は有効であることがわかった。有効であった要因として考えられるの

は、節選択モデルに対する入力として、節のタイトルをそれぞれ個別で入力するのではなく、すべての節のタイトルを結合して入力を与えたことであると思われる。節のタイトルを結合することで、節の前後関係を含めて与えた際に、実際に節選択モデルに使われている Transformer が前後関係を利用できていたからであると思われる。Transformer における入力では、Positional Encoding によって位置情報がトークン単位で与えられ、入力時にトークンごとの埋め込み表現に加えられるため、節の選択を行う際に、前後関係が情報として与えられていることとなる。そのため、たとえば序論の次に来る節は関連知識であると推論したり、実験の節の前の節は提案手法の節であると推論することが可能になったと思われる。つまり、節のタイトルを全て結合し、Transformer に与えることで、節のタイトル名と前後情報の 2 つで、要約に必要な節の選択を行うことが可能になったと思われる。

しかし、節選択モデルでも十分な節選択ができていないことも確認できた。生成された要約の中には提案手法の内容で生成が終了されているものも確認された。そこで、節選択モデルが節選択を行った際、必要な節をどれだけ選択できていたのか評価を行った。評価用のデータセットに対して、5.4.1 にて述べた方法で、要約に必要な節を選択できたかどうかの評価用のデータセットを構築した。この評価用データセットを用いて評価を行ったところ、平均して 2 つほど節の選択がされていない、もしくは余分に選択されていることが判明した。つまり、現状の節選択モデルを用いて要約生成を行った場合、情報が不足する、もしくは冗長な要約が生成される場合がある。

このような問題を解決する方法としてまず考えられるのは、few-shot 学習用データセットの数を増やすことであるが、現在公開されている論文要約用のコーパスは自然言語処理学会論文誌コーパスのみであるため、これ以上増やすことは難しいと思われる。次に考えられるのは、節選択モデルの構築における損失関数の変更である。現状、節選択モデルの構築に用いている損失関数は要約生成モデルと同様の負の対数尤度である。ここに、選択した節の数と正解データの節の数の差に対するペナルティを与えることで、選択する節の数の差が無くなるように学習することが期待される。もう一つの方法として、与えるデータの中で選択する節の数を明示するという方法が考えられる。これは、タスク名を入力データ内で明記する T5 モデルの特徴を利用した方法である。入力データの中には 5.4.1 において示したとおり、タスク名と要約対象の論文の節のタイトル名が含まれているが、この場合、与えられる情報としてタスクの内容と節のタイトル名一覧とその前後関係だけである。そのため、入力情報には選択しなければならない節の数も与えられていない。これを解消することで、学習時や選択時に使用できる情報の数を増やすことで、精度が高まることが期待される。

7.3.2 要約生成モデルと学習データ

本研究の提案手法において、要約生成モデルは DANCER に基づいて学習が行われている。DANCER では、論文自動要約モデルにおける学習で、節ごとに要約を生成するように学習する

ため、全文入力による要約生成モデルよりも学習データ数は増加する。そのため、学習データ数不足による生成された要約文内における単語やフレーズのリピートの発生を抑えることができる。全文入力における要約生成ではフレーズや単語の繰り返しが多発していたが、提案手法の要約生成モデルでは抑えられていた。しかし、DANCER に沿って学習したモデルにおいても同様の要約生成が確認された。このように、DANCER によって学習データ数を増加させることでリピートの発生を抑えることはできたが、完全ではない。つまり、要約生成モデル構築のための学習データ数が DANCER を用いても足りなかったと推測できる。

この問題の解決方法として考えられるのは、論文要約用のデータセット以外の要約用データセットを学習データとして用いる方法である。DANCER による学習では論文の要旨の各文を、内容が類似している節に割り当てるため、学習データ数の増加とともに、学習に用いる要約の長さも短くなる。そのため、論文よりも短い長さで要約をニュース記事など、他分野の要約データセットを追加の学習データとして使用できると考えられる。論文要約という分野に限定しなければ、現在日本語向けの要約用データセットとして、次の 2 つが存在している。

- wikihow データセット [52]
- livedoor ニュース 3 行要約データセット [53]

wikihow とは、ウィキ型の様々なハウツーについてまとめられた大型サイトであり、専門家によるハウツー記事や、ユーザーのハウツー記事など多く投稿されている。このサイトのハウツー記事では、1 つの記事につき、複数の方法が紹介されており、方法 1 つごとに対応した手順が書かれている。この手順の見出しを要約、手順の詳細を本文として要約用データセットを構築する。日本語用データセットも公開されており、データ数は 5066 件と数が非常に多い。livedoor ニュースとは、大型のニュースポータルサイトであり、このニュースサイトで公開されている多くの記事には、3 文で構成される要約が人手によって付与されている。livedoor ニュース 3 行要約データセットは 2014 年 1 月から 2016 年 12 月までに公開されたニュース記事が含まれており、215560 件のデータからなる。これら 2 つのデータセットを合わせれば、凡そ 23 万件となり、学習データとして十分な数を確保することができると思われる。別の方法として、既存の英語コーパスを日本語に翻訳して利用するという方法も考えられるが、この場合、翻訳モデルの翻訳精度の影響を受けてしまうと考えられる。しかし、arXiv コーパスや PubMed データセットなどの大型データセットを利用できるため、有用であると思われる。

7.3.3 他言語への提案手法の適用について

本研究の提案手法は節のタイトルから、必要な節を選択し、それらの節から要約を生成し、結合する。また、ニューラルネットワークによる要約生成のため、品詞や文法などの言語の特性の影響は受けにくい。そのため、日本語だけでなく、データさえあれば他の言語でも同様に有効であると

考えられる。しかし、節のタイトル名において英語と日本語の差異が存在する。日本語の場合、著者独自のタイトル名を設定することが多いが、英語の場合、タイトル名は統一されていることが多く、著者ごとに大きな違いはない。例えば、日本語の場合、手法については著者ごとに独自のタイトルを設定することが多いが、英語の場合、Method や Model など単語のみで簡潔に設定することが多い。これは例に挙げた手法以外にも当てはまり、実験や結果や導入など、多くの場合において統一されたタイトル名を付与している。そのため、タイトル名とその前後関係からどの節が要約に必要な節なのか判定することが難しいと考えられる。ただし、独自にタイトル名をつける傾向にあるような言語の論文であれば、本手法や有効に働くと思われる。

7.3.4 他分野論文への応用

本研究では要約生成モデルと節選択モデルの構築には自然言語処理の分野に関係する論文のみを適用して行ったため、それ以外の分野においては有効なのかどうかというのは不明である。しかし、多くの論文は同様の構造を持っているため、学習データさえ確保できれば他の研究分野においても適用可能であり、実験の結果と同様の結果を発揮すると思われる。ただし、分野ごとに論文における節のタイトル名のつけ方において、異なる特徴を持つため、分野ごとに論文の構造を確かめる必要がある。そのため、実際にどれほどの性能を発揮するのかどうか、今後の課題として確かめる必要がある。

7.4 提案手法の応用

7.4.1 提案手法の要約分野への応用について

本研究の提案手法は、論文の自動要約という点に絞った手法ではあるが、類似の構造を持つ文書であれば、関連論文 (2) において確認されている。関連論文 (2) にて、電気通信大学の入学試験において、英語科目にて出題される小論文を対象に序論と結論に該当する箇所から抽出型要約を行った結果、全文から要約を生成する場合よりも ROUGE-L の結果が高くなった。このことから、論文と類似した文書構造を持つのであれば、本研究の提案手法による要約生成が有効であると考えられる。小論文や評論文のような文書以外にも適用先として考えられるのは、節構造を持つ説明書の要約であると思われる。説明書であれば、手順と、その手順の操作でもたらされる結果といった、構造的ではあるが、読むのに非常に時間が掛かる文書においては効果を発揮すると思われる。対して、小説のような長文だが節の構造を持たない、もしくは節の前後関係が一定ではないような長文の文書においては、提案手法はあまり有効でないと思われる。例えば、小説の場合では、先に未来の話を書いて、次に過去の話を書き、そのあとに現在の話を書く、また、別の順番で話を展開するなど、多種多様な書き方があるため、節の前後関係による要約対象の選択を行うのは困難であると思われる。

また、論文の複数文書要約として応用するという事も考えられる。論文に対する複数文書要約は過去、コンペティション等で行われている [54]。論文における複数文書要約とは、複数の論文を情報源として、それらの論文の分野における技術や研究の動向について要約を行うというものである。例えば、文章生成という分野に関係する論文を集めて、複数文書要約を行った場合、Transformer についてまとめた要約や、chatGPT についてまとめた要約の生成を行う。今現在、数多くの論文が投稿されており、それらの流行を追うには複数の論文を読み込み、要約し、まとめる必要がある。そのため、人手で行うには労力のかかる作業であるため、自動化が望まれている分野である。本研究の提案手法をこの複数論文要約に適用する場合、提案手法は要旨の文ごとに節を割り当てるため、割り当てに使う要旨が、複数文書要約の正解データに変わる。正解データに含まれている各文がまず、どの論文の要約に該当するのか選択を行い、さらに該当した論文内のどの節に該当するのか選択を行う、複数階層の形式で節選択を行うことで応用することが可能になると考えられる。

要約作成の支援として応用するならば、要旨の執筆支援システムへの転用が考えられる。本研究の提案手法は節のタイトルを元にどの節が要旨に含まれているべきかを判定する。これを利用することで、どの節を要旨にふくめたらよいかどうか提案することが可能となる。さらに、実際に執筆した要旨と本研究の提案手法による生成した要旨と比較することで、どの節が含まれていないのか、など評価を行うことも可能となると思われる。他にも、節選択モデルのみを用いることで、論文を読む際に最低限必要な節を提示することも可能となる。これらの実現には提案手法の節選択モデルを用いることで対応できる。節選択モデルに対し、読みたい論文の節タイトルを一覧を入力させることで、要旨に含まれているであろう節、つまりは重要な節の一覧を出力させることが可能である。また、執筆支援としての適用には節選択モデル構築時に使用する学習データの作成方法である DANCER を適用することで実現が可能である。これらの応用は、論文と同一構造を持ち合わせている小論文や評論においても適用することが可能であると実験の結果より、考えられる。

7.4.2 要約分野以外への応用

本研究の提案手法、特に節選択モデルは要約分野以外への応用が可能であると考えられる。節選択モデルの構築には、要約文、選択の対象となる節と節の内容が必要であり、節の内容と要約文は何らかの対応関係にある必要がある。しかし、対応関係があるものであれば、様々なものに置換可能であり、要約以外の分野へ応用することが可能である。

まず、教育分野への応用である。現在、平成 31 年の学習指導要領の変更に伴い、来年度以降は情報科目も共通テストに含まれることになり、現在の高校生における学習の負担はより大きいものになると考えられる。また、共通テストのみならず、大学入試までを見据えた学習となると、共通テストにおける全ての科目において全ての内容を網羅的に学習することは非常に困難である。

また、過去の共通テストを用いて勉強を行うとしても、年度ごとに出題される内容というのは異なるため、どの内容を勉強するのが良いのか簡単に判断はできない。そこで、共通テストの問題を節選択モデルにおける要約文、各科目にて学習する内容を節タイトルとして取り扱うことで、共通テスト内の問題と科目内容を紐づけさせた選択モデルの構築が可能となる。これにより、共通テストの問題に出てくる内容に限定して取り組むことが可能となり、高校生の学習において大学入試に必要な科目へ専念するための時間の確保など、負担を減らすことにつながるのではないかと考えられる。実現のためには、問題と科目内の内容を対応付けた学習データの作成が必須となるが、模試においてはどの問題がどの分野に対応しているのかについての情報が公開されることが多いため、これを用いることで学習データの構築が可能になると思われる。

また、逆に特定分野の学習を行う際の参考書のレコメンドにも応用可能である。ある分野の内容を要約文、書籍のタイトルを節タイトル、書籍の内容を節本文とすることで、要約文の内容を多く含んでいる節本文、つまりは分野の内容を網羅している書籍を選択する選択モデルの構築を行うことが可能である。この選択モデルの利点は、特定の1冊をレコメンドするだけではなく、複数の書籍をレコメンドすることも可能であるため、レコメンド後に比較検討を行ったり、複数の資料として学習を行うなども可能である。さらに、好きな本や映画のキーワードを要約文、タイトルと内容を節及び節本文とすることで、学習用参考書以外のレコメンドも可能になるのではないかと考えられる。また、学習元となる本の内容や購入傾向についてはECサイトなどのレビューが書かれているサイトから取得が可能であるため、学習データ構築は容易であると考えられる。

他にも、旅行での観光名所をまわる際の選択においても応用可能であると考えられる。国内外問わず、観光名所というのは地域ごとに多く存在しており、どこを回るのが良いのか考えるのにはコストを要する。そこで、節選択モデルを次のように応用する。例えば、旅行の目的を要約文、観光名所の概要を節本文、観光名所を節タイトルとすることで、回るべき観光名所のリストをピックアップしてくれる選択モデルとなる。これを1日目の目的、2日目の目的と複数用意することで、旅行計画を組み立てる支援ともなる。また、選択モデルの出力時の出力順を、泊まる予定のホテルから近い順にする、最寄りの駅順にするなど、ルート構築の参考にすることもできる。これらを実現するためには、観光名所に対する評価データと距離のデータが必要となるが、評価データはレビューサイトから、距離データは現在では様々なところから取得することが可能であるため、学習データの構築については問題はないと思われる。

付録: 節選択モデルが選択した節一覧

第5章の検証にて、節選択モデルが選択した節の一覧を示す。比較として、節選択モデル用の学習データ構築と同様の手法で選択された節を正解データとして示す。

	タイトル	節一覧	正解データ	選択されたもの
0	大語彙を対象とした音声対話インタフェースにおける自然な応答生成	”はじめに, 大語彙音声対話インタフェースの課題, 思い込み戦略, 認識エンジンを用いた思い込み対象の分析, 大語彙インタフェースの実装, まとめ, 及び今後の課題”	”はじめに, 大語彙音声対話インタフェースの課題, 認識エンジンを用いた思い込み対象の分析”	”はじめに, 大語彙インタフェースの実装”
1	実世界参照による分野特有の固有表現認識の精度向上	”はじめに, 関連研究, 将棋解説コーパス, 提案手法, 評価実験, おわりに”	”関連研究, 将棋解説コーパス, 評価実験”	”はじめに, おわりに”
2	質問応答に基づく対災害情報分析システム	”はじめに, 質問応答に基づく対災害情報分析システム, システムの評価実験, さらに行き届いた被災情報の活用を目指して, 関連研究, おわりに”	”はじめに, さらに行き届いた被災情報の活用を目指して”	”はじめに, システムの評価実験, 関連研究, おわりに”
3	文字列正規化パターンの獲得と崩れ表記正規化に基づく日本語形態素解析	”はじめに, 関連研究, 提案手法, 実験, まとめと今後の課題”	はじめに	はじめに
4	「ロボットは東大に入れるか」プロジェクト: 代ゼミセンター模試タスクにおけるエラーの分析	”はじめに, センター試験タスクの概要, 英語問題のエラー分析, 国語 評論問題のエラー分析, 数学問題のエラー分析, 物理問題のエラー分析, 世界史・日本史のエラー分析, おわりに”	”はじめに, センター試験タスクの概要, 英語問題のエラー分析, 数学問題のエラー分析, 世界史・日本史のエラー分析”	”はじめに, 世界史・日本史のエラー分析”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
5	かな漢字換言を通じた日本語語義曖昧性解消の分析	”はじめに, 関連研究, かな漢字換言, 提案手法, 実験, 結論”	”はじめに, 提案手法”	”はじめに, 結論”
6	日本語の照応関係理解に関する一考察—「主題」(Topic)が果たす役割を中心に—	”はじめに, 人間の照応関係理解, 主題割当方略 (Topic Assignment Strategy), 主語割当方略および平行機能方略, 発見的ストラテジー間の相互関係, まとめと今後の課題”	”はじめに, 主題割当方略 (Topic Assignment Strategy), 主語割当方略および平行機能方略, 発見的ストラテジー間の相互関係”	”はじめに, 発見的ストラテジー間の相互関係”
7	統語ベース翻訳のための構文解析器の自己学習	”はじめに, Tree-to-String 翻訳, 構文解析の自己学習, 統語ベース翻訳のための構文解析器の標的自己学習, 評価, おわりに”	”はじめに, 構文解析の自己学習, おわりに”	”はじめに, Tree-to-String 翻訳, 構文解析の自己学習, 評価”
8	後方文脈を考慮した係り受けモデル	”はじめに, 係り受け確率モデル, 実験結果, おわりに”	はじめに	はじめに
9	連想メカニズムを用いた話者の感情判断手法の提案	”はじめに, 感情判断システム, 連想メカニズム, 対象語の処理, 変化語の処理, 感情判断の処理, 感情判断システムの性能評価, おわりに”	”はじめに, 対象語の処理, 感情判断の処理”	”はじめに, 感情判断システム, 連想メカニズム, 対象語の処理, 変化語の処理, 感情判断の処理, 感情判断システムの性能評価”
10	「コーパスベース国語辞典」構築のための「古風な語」の分析と記述	”はじめに, 「古風な語」の現行辞典での扱い, 「古風な語」の従来の国語辞典における記述, 「古風な語」のコーパス分析, コーパス分析を活かした辞書記述, おわりに”	”はじめに, おわりに”	”はじめに, 「古風な語」の従来の国語辞典における記述, コーパス分析を活かした辞書記述”
11	カテゴリ名と記事名の意味属性分類に基づく Wikipedia からの上位下位関係オントロジーの構築	”序論, オントロジーと Wikipedia, 汎用オントロジー構築手法, 実験条件, 実験結果, 考察, 関連研究, 結論”	”序論, 関連研究”	”序論, 考察, 関連研究”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
12	動詞訳語選択のための「格フレーム木」の統計的な学習	”はじめに, 人手による格フレーム獲得の問題点, 対訳コーパスからの格フレーム木の学習, 英日対訳コーパス, 語形を利用した実験, 意味コードを用いた実験, むすび”	”はじめに, 対訳コーパスからの格フレーム木の学習, 英日対訳コーパス, 語形を利用した実験”	”はじめに, 人手による格フレーム獲得の問題点, 対訳コーパスからの格フレーム木の学習, むすび”
13	文書拡張によるキーワード抽出	”はじめに, キーワードの定義, キーワード抽出, 文書拡張による反復度と出現確率のヒストグラム, キーワード抽出実験, 先行研究との比較, 結論”	”はじめに, 文書拡張による反復度と出現確率のヒストグラム, 先行研究との比較”	”はじめに, キーワード抽出, キーワード抽出実験, 先行研究との比較”
14	コンパラブルコーパスと対訳辞書による日英クロス言語検索	”はじめに, キュエリタームの訳語選択, GDMAX 訳語選択法, 実験・評価, 考察, 関連研究, おわりに”	”はじめに, GDMAX 訳語選択法, 実験・評価, 考察”	”はじめに, キュエリタームの訳語選択, GDMAX 訳語選択法, 考察, 関連研究”
15	最大エントロピーモデルに基づく形態素解析 qp —未知語の問題の解決策—	”はじめに, 形態素モデル, 実験と考察, 関連研究, まとめ”	”はじめに, 関連研究”	はじめに
16	文書間類似度について	”はじめに, 評価指標と距離・類似度・カーネル・順序尺度・相関係数, 評価に用いる言語資源, 尺度の定性的な分析, おわりに”	”はじめに, 尺度の定性的な分析”	”はじめに, 評価指標と距離・類似度・カーネル・順序尺度・相関係数, 評価に用いる言語資源, おわりに”
17	文書走査を用いた複合名詞解析	”はじめに, 複合名詞解析の構成, 従手法と問題点の分析, 文書走査による複合名詞解析, 実験結果, 考察, おわりに”	”はじめに, 複合名詞解析の構成, 実験結果, 考察”	”はじめに, 考察”
18	クラスタリングを利用した語義曖昧性解消の誤り原因のタイプ分け	”はじめに, 分析対象データ, 各人の分析結果, クラスタリングを用いた分析結果の統合, 考察, おわりに”	”各人の分析結果, クラスタリングを用いた分析結果の統合”	”はじめに, 分析対象データ, 各人の分析結果, クラスタリングを用いた分析結果の統合, 考察”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
19	テキスト自動要約に関する研究動向(巻頭言に代えて)	”はじめに, 重要文抽出による要約手法, 抽象化, 言い換えによる要約手法, ユーザに適応した動的な要約手法, 複数テキストを対象にした要約手法, 文中の重要箇所抽出, 不要箇所削除による要約手法, 要約の表示方法について, 要約の評価方法について, おわりに”	”はじめに, 抽象化, 言い換えによる要約手法, 文中の重要箇所抽出, 不要箇所削除による要約手法”	”はじめに, 重要文抽出による要約手法, 文中の重要箇所抽出, 不要箇所削除による要約手法”
20	セグメントの分割と統合による文章の構造解析	”はじめに, 文章の論説モデル, 文章解析のトップダウン的アプローチ, 文章解析のボトムアップ的アプローチ, トップダウン的アプローチとボトムアップ的アプローチの融合, 解析システムと実験, まとめ”	”はじめに, 文章解析のボトムアップ的アプローチ”	”はじめに, まとめ”
21	語彙的連鎖に基づくパッセージ検索	”はじめに, パッセージレベルの文書検索, 語彙的連鎖の計算, 語彙的連鎖に基づくパッセージ検索, 実験, おわりに”	”はじめに, パッセージレベルの文書検索, おわりに”	”はじめに, パッセージレベルの文書検索, 語彙的連鎖の計算, 実験, おわりに”
22	発話タイプ付きコーパスを用いた確率的対話モデルの自動生成	”はじめに, IFT 付きコーパス, エルゴード HMM による対話構造のモデル化, 状態マーキング手法による対話構造のモデル化, おわりに”	”はじめに, おわりに”	はじめに
23	被喩詞の意味と比喩表現の意味との違いを示す指標	”はじめに, 情報量を用いた比喩の定式化, SD 法による比喩表現の意味の測定, おわりに, 付録”	”はじめに, 情報量を用いた比喩の定式化”	”はじめに, おわりに, 付録”
24	日本語-ウイグル語辞書の半自動作成と評価	”はじめに, ウイグル語-日本語辞書の電子化, 日本語-ウイグル語電子辞書の自動生成, 日本語-ウイグル語辞書の翻訳システム用辞書への変換, 日本語-ウイグル語電子辞書の評価, まとめ”	”はじめに, ウイグル語-日本語辞書の電子化, 日本語-ウイグル語辞書の翻訳システム用辞書への変換”	”はじめに, まとめ”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
25	選択肢が選択または排除されるきっかけの理由を Elimination-By-Aspects で捉える	”はじめに、関連研究、データ収集方法、EBA に則った取捨選択プロセスの記述、行間を読み取る、選択または排除されるきっかけ、おわりに”	”はじめに、関連研究、データ収集方法、行間を読み取る”	”はじめに、関連研究、データ収集方法、行間を読み取る、選択または排除されるきっかけ”
26	“名詞 A のような名詞 B” 表現の比喩性判定モデル	”はじめに、“名詞 A のような名詞 B” 表現のパターン分類、比喩性判定モデルの提案、コーパスでの検証、比喩語の存在、おわりに”	”はじめに、おわりに”	”はじめに、おわりに”
27	対話行為に固有の特徴を考慮した自由対話システムにおける対話行為推定	”はじめに、関連研究、提案手法、評価実験、おわりに”	”はじめに、関連研究、提案手法、評価実験、おわりに”	”はじめに、関連研究、提案手法、評価実験、おわりに”
28	動詞待遇表現に対する丁寧さの印象に関する定量的分析 - 接頭辞オを用いた表現と接頭辞ゴを用いた表現との比較 -	”はじめに、待遇表現の丁寧さの定量化、オ～型表現とゴ～型表現の丁寧さに関する実験、実験結果、考察、おわりに”	”はじめに、オ～型表現とゴ～型表現の丁寧さに関する実験、考察”	”はじめに、待遇表現の丁寧さの定量化”
29	言語処理技術と教材作成の連携 - データベース・ソフトウェアを用いた英語学習教材の自動作成 -	”はじめに、データベース・ソフトウェアについて、連携事例 I: Phrase Reading 教材の自動作成、連携事例 II: Cloze テストの自動作成、おわりに”	”はじめに、データベース・ソフトウェアについて”	はじめに
30	機械学習手法を用いた名詞句の指示性の推定	”はじめに、名詞句の指示性の分類、名詞句の指示性の推定方法、実験と考察、おわりに”	”はじめに、名詞句の指示性の推定方法、実験と考察”	”はじめに、おわりに”
31	大域的な並べ替えによるサブ言語翻訳の高精度化	”はじめに、関連研究、大域的並べ替え手法、評価実験、実験結果、考察、おわりに”	”はじめに、大域的並べ替え手法、評価実験、実験結果”	”はじめに、実験結果”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
32	多段解析法による形態素解析を用いた音声合成用読み韻律情報設定法とその単語辞書構成	”はじめに, テキスト音声合成の流れ, 形態素解析における読み韻律情報設定のための特徴, 読み韻律情報設定のための単語辞書情報, 読み韻律情報の設定法, 評価, おわりに”	”はじめに, テキスト音声合成の流れ, 評価”	”はじめに, テキスト音声合成の流れ, 形態素解析における読み韻律情報設定のための特徴, 読み韻律情報の設定法, 評価, おわりに”
33	二言語コーパスからの語彙知識獲得のための対訳辞書登録候補の選別	”はじめに, 着目した素性, 訓練事例集, 実験と考察, おわりに”	”はじめに, 着目した素性, 訓練事例集, 実験と考察”	”はじめに, おわりに”
34	文脈限定 Skip-gram による同義語獲得	”序論, 関連研究, 提案手法, 使用データと予備実験, 実験結果, 結論”	結論	”序論, 関連研究, 提案手法, 実験結果”
35	対話者の社会的役割を利用した訳し分け手法	”はじめに, 『話し手の役割』と『丁寧さ』, 話し手の役割情報を翻訳知識に組み込む方法, 実験, 考察, おわりに”	”はじめに, 『話し手の役割』と『丁寧さ』, おわりに”	”はじめに, 話し手の役割情報を翻訳知識に組み込む方法, 考察, おわりに”
36	語彙的連鎖に基づく要約の情報検索タスクを用いた評価	”はじめに, 語彙的連鎖型パッケージ抽出法に基づく要約, 評価実験, 結果と考察, おわりに”	”はじめに, 評価実験, 結果と考察”	”はじめに, 語彙的連鎖型パッケージ抽出法に基づく要約, 評価実験, 結果と考察, おわりに”
37	順接複文における主語の共参照関係の分析	”はじめに, 順接複文の性質, IPAL の述語の素性, 例文の分析, おわりに”	”はじめに, おわりに”	”はじめに, おわりに”
38	構造化された言語資料に対する全文検索システムの設計と実現	”はじめに, 前提条件, システムの概要, 検索方式, 評価, 関連研究との比較, おわりに”	”はじめに, 関連研究との比較, おわりに”	”はじめに, 前提条件, おわりに”
39		”はじめに, 関連研究, データと問題設定, データベースフィールドへの分類および根拠抽出手法, 評価実験, 結論”	”はじめに, データと問題設定, 評価実験”	”はじめに, 評価実験”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
40	記述された「併置型駄洒落」の音素上の性質	”まえがき, 分析のための準備, 併置型駄洒落の音素上の分析, 考察と課題, むすび, 謝辞, 付録: 基準辞書について”	”まえがき, 考察と課題”	”まえがき, 付録: 基準辞書について”
41	言い換えと逆翻字を用いた片仮名複合名詞の分割	”はじめに, 関連研究, 教師あり学習に基づく手法, 言い換え素性, 逆翻字素性, 実験と議論, おわりに”	”はじめに, 教師あり学習に基づく手法, 逆翻字素性”	”はじめに, 逆翻字素性”
42	トーナメントモデルを用いた日本語係り受け解析	”はじめに, トーナメントモデル, 議論, 評価実験, 議論と今後の課題, まとめ”	”はじめに, トーナメントモデル, 議論”	”はじめに, トーナメントモデル, 議論, 議論と今後の課題, まとめ”
43	看護学分野の専門用語抽出方法の研究	”はじめに, 従来研究とアプローチ, 提案手法, 実験及び評価, 考察と今後の課題, まとめ”	”はじめに, 考察と今後の課題”	”はじめに, 従来研究とアプローチ, 提案手法”
44	日中機械翻訳におけるとりたて表現の翻訳について – 「も」, 「さえ」, 「でも」 –	”はじめに, とりたて表現の特徴と中国語との対応関係, 取立て表現の中国語への翻訳, 3 つのとりたて詞の翻訳手順, 翻訳規則の評価と問題, おわりに”	”はじめに, とりたて表現の特徴と中国語との対応関係, 3 つのとりたて詞の翻訳手順, おわりに”	はじめに
45	述語と項の位置関係ごとの候補比較による日本語述語項構造解析	”はじめに, 関連研究, 述語と項の位置関係ごとの候補比較による日本語述語項構造解析, 評価実験, 議論, 事例分析, おわりに”	”はじめに, 評価実験, 事例分析”	”はじめに, 事例分析”
46	3つ以下の候補から係り先を選択する係り受け解析モデル	”はじめに, 関連研究, 本研究の手法, 実験結果, 考察, まとめ”	はじめに	はじめに
47	サポートベクタマシンを使った文書分類における 仮想事例の利用	”はじめに, サポートベクタマシン, 仮想事例と仮想サポートベクタ, 文書分類のための仮想事例, 評価実験と議論, 関連研究との比較, おわりに”	”文書分類のための仮想事例, 評価実験と議論, 関連研究との比較”	”はじめに, 仮想事例と仮想サポートベクタ, 評価実験と議論, 関連研究との比較”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
48	「怒り」の発話を対象とした話者の感情の程度推定法	”はじめに, 関連研究, 音声資料, 「怒り」を含む発話の分析, 推定実験, おわりに”	”はじめに, 関連研究, 「怒り」を含む発話の分析”	はじめに
49	頑健な英日機械翻訳システム実現のための原文自動前編集	”はじめに, 書き換え対象文, 原文書き換え系, 実験と考察, 関連研究との比較, おわりに”	”はじめに, 書き換え対象文, 関連研究との比較, おわりに”	”はじめに, 書き換え対象文, 原文書き換え系, 関連研究との比較, おわりに”
50	チャット対話における発言間の継続関係と応答関係の同定	”はじめに, 対話構造のモデル, 実験用コーパス, 提案手法, 評価実験, 関連研究, おわりに”	”はじめに, 対話構造のモデル, 提案手法”	”はじめに, 提案手法, 評価実験”
51	文体と難易度を制御可能な日本語機能表現の言い換え	”はじめに, 2つの階層構造を持つ機能表現辞書, 本論文で提案する機能表現の言い換え手法, 機能表現言い換えシステム, 評価, 関連研究, おわりに”	”はじめに, 本論文で提案する機能表現の言い換え手法, 機能表現言い換えシステム, 評価”	”はじめに, 評価”
52	中間言語を用いたインドネシア語-日本語対訳辞書の拡充	”はじめに, 中間言語を用いた対訳辞書の拡充, 評価実験, 関連研究, むすび”	”はじめに, 評価実験, 関連研究”	”はじめに, 評価実験, むすび”
53	英日機械翻訳における自然な和文生成のための英語名詞句の書き換え	”はじめに, 書き換える分類, 名詞句の書き換え, 実験と考察, 関連研究, おわりに”	”はじめに, 関連研究”	”はじめに, 名詞句の書き換え, 関連研究”
54	歴史的日本語資料を対象とした形態素解析	”はじめに, 研究の背景, 見出し語の追加と学習用コーパスの作成, 解析精度の評価, エラー分析, おわりに”	”はじめに, 研究の背景, エラー分析”	”はじめに, 研究の背景”
55	要約事例を用例として模倣利用したニュース記事要約	”はじめに, 用例利用型のアプローチ, 提案法のシステム概要, 類似用例文の選択, 文節の対応付け, 文節の組合せ, 実験, 結果及び考察, 結論”	”はじめに, 用例利用型のアプローチ, 提案法のシステム概要, 類似用例文の選択”	”はじめに, 用例利用型のアプローチ, 提案法のシステム概要, 類似用例文の選択”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
56	文法性・流暢性・意味保存性に基づく文法誤り訂正の参照無し評価	”序論, 自動評価尺度の評価方法, 既存の評価尺度, 提案手法, 実験, 結論”	”序論, 自動評価尺度の評価方法, 提案手法, 実験, 結論”	序論
57	構文解析と融合した階層的句アライメント	”はじめに, 階層的句アライメントの基本方式, 句アライメントと構文解析の融合, 単語アライメントに要求される機能, 実験・評価, 関連研究, まとめ”	”はじめに, 階層的句アライメントの基本方式, 実験・評価”	”はじめに, 階層的句アライメントの基本方式, 句アライメントと構文解析の融合, 実験・評価, 関連研究”
58	人間による翻訳文と機械翻訳文における動詞の馴染み度の比較分析	”はじめに, 分析方法, 分析結果, おわりに”	”はじめに, 分析方法”	”はじめに, 分析結果”
59	社会学における職業・産業コーディング自動化システムの活用	”はじめに, 自動化システムの変遷, 職業・産業コーディング自動化システム, システムの評価, システムの利用方法, 関連研究, おわりに”	”はじめに, 自動化システムの変遷, システムの評価, おわりに”	”はじめに, 自動化システムの変遷, 関連研究, おわりに”
60	ニューラルネットワークを利用した中国語の統合的な構文解析	”はじめに, 関連研究, モデル, 実験, 将来研究, 結論”		”はじめに, 関連研究, モデル, 実験, 将来研究”
61	日本語を援用した日本手話表記法の試み	”はじめに, 手話について, 手話表記法の提案, 手話映像の書き取り実験, 関連研究, おわりに”	”はじめに, 手話表記法の提案, 手話映像の書き取り実験, おわりに”	はじめに
62	シソーラスを組み込んだ意味解析システム	”はじめに, 構文構造の決定, 係り受けデータの整理, 情報の付与, 辞書, 結果, おわりに”	”構文構造の決定, 係り受けデータの整理, 情報の付与, 辞書”	”はじめに, 係り受けデータの整理, 辞書”
63	文脈素性のベクタ空間モデルを用いた日英翻訳選択	”はじめに, 文脈素性ベクタを用いた翻訳選択, 文脈素性種別と翻訳選択性能との関係, まとめ”	はじめに	はじめに

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
64	訂正パターンに基づく誤情報の収集と拡散状況の分析	”はじめに, 関連研究, 提案手法, 実験, 誤情報の拡散状況の分析, おわりに”	”はじめに, 関連研究, 実験, 誤情報の拡散状況の分析”	”はじめに, 誤情報の拡散状況の分析”
65	点予測による形態素解析	”はじめに, 点予測を用いた形態素解析, 評価, おわりに”	”はじめに, おわりに”	はじめに
66	Twitter を用いた東日本大震災時の首都圏の帰宅意思決定分析	”はじめに, Tweet Data Behavioral Data, 帰宅行動要因の分析, 行動データから意思決定モデルの構築, 推定結果とシミュレーション, おわりに”	”はじめに, 帰宅行動要因の分析, 推定結果とシミュレーション, おわりに”	はじめに
67	言語横断質問応答に適した機械翻訳評価尺度の調査	”はじめに, 本調査の概観, データセット作成, 質問応答システム, 実験, まとめ”	”質問応答システム, 実験”	”はじめに, 本調査の概観, データセット作成, 質問応答システム, 実験, まとめ”
68	情報検索システムの統計的手法による特徴と精度の分析	”はじめに, 再現率と適合率の関係, 評価実験, まとめ”	”再現率と適合率の関係, 評価実験, まとめ”	”はじめに, 再現率と適合率の関係, 評価実験”
69	確率的クラスタリングを用いた文書連想検索	”はじめに, クラスタ検索, 確率的クラスタリング (HBC), 実験, おわりに”	はじめに	はじめに
70	多重タグ付き英語学習者コーパスの開発と 英語能力自動測定への応用	”はじめに, 学習者コーパスの収集方法, 学習者コーパスの分析, 訳質自動評価への応用, まとめと今後の課題”	”はじめに, 学習者コーパスの分析, 訳質自動評価への応用”	”はじめに, 訳質自動評価への応用”
71	事象に対する網羅的な時間情報アノテーションとその分析	”はじめに, 関連研究, アノテーション基準, アノテーション結果, アノテータ間でのタグの揺れ分析, 時間情報推定, 本アノテーションの応用, おわりに”	”はじめに, 関連研究, アノテーション基準, アノテーション結果, 時間情報推定, 本アノテーションの応用”	”はじめに, おわりに”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
72	NMF による重み付きハイパーグラフを用いた アンサンブル文書クラスタリング	”はじめに, NMF と初期値の問題, アンサンブルクラスタリング, 実験, 考察と関連研究, おわりに”	”アンサンブルクラスタリング, 実験, 考察と関連研究, おわりに”	”はじめに, おわりに”
73	講演の書き起こしに対する統計的手法を用いた文体の整形	”緒論, 整形作業における処理, 統計的手法による文体の整形, 実験と評価, 結論”	緒論	緒論
74	英日機械翻訳における代名詞翻訳の改良	”はじめに, 代名詞の日英対照比較, 決定木学習の利用, 正解付きコーパスの作成, 着目した属性, 実験と考察, おわりに”	”はじめに, 決定木学習の利用, 正解付きコーパスの作成, 実験と考察”	”はじめに, おわりに”
75	確率的言語モデルに基づく多言語コーパスからの言語系統樹の再構築	”はじめに, 従来の研究, 確率モデルに基づく言語のクラスタリング, 評価実験, おわりに”	”はじめに, おわりに”	”はじめに, 従来の研究, 確率モデルに基づく言語のクラスタリング, おわりに”
76	自然言語の構文解析のための LR 解析表の圧縮法	”はじめに, LR 解析表の圧縮, 実装と実験, 更なる圧縮のための改良手法, 関連研究, 結論”	結論	”はじめに, LR 解析表の圧縮, 結論”
77	文字間統計情報に基づく口語文字列の自動抽出	”はじめに, 日本語における語句抽出, システム概要, 実験, まとめ”	”はじめに, 日本語における語句抽出, システム概要, 実験, まとめ”	”はじめに, まとめ”
78	評判情報のレベルを考慮した評価文書の分類と 評価情報の信頼性評価への応用	”はじめに, 評判情報, 評価文書の分類, 評価情報の信頼性評価, 評価実験, おわりに”	”はじめに, 評価実験”	”はじめに, 評価実験”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
79	情報検索のための表記の揺れに寛容な類似尺度	”はじめに, 編集距離の類似尺度への変換とその拡張, 本論文で用いる重み, 実験の概要, 編集類似度の性質および検索性能の検証, 基本的な類似尺度との検索性能比較, 考察, まとめ”	”はじめに, 編集距離の類似尺度への変換とその拡張, 本論文で用いる重み”	”はじめに, 編集距離の類似尺度への変換とその拡張, 本論文で用いる重み, 編集類似度の性質および検索性能の検証”
80	集合間類似度に対する簡潔かつ高速な類似文字列検索アルゴリズム	”はじめに, 類似文字列検索の定式化, データ構造とアルゴリズム, 実験, 関連研究, まとめ, 付録: その他の類似度関数の条件式の導出”	”はじめに, 類似文字列検索の定式化, 実験”	”はじめに, 関連研究”
81	機械学習による複数文書からの重要文抽出	”はじめに, 対象とする複数文書, Support Vector Machine に基づく複数文書からの重要文抽出手法, 評価実験, 冗長性削減の有効性, まとめ”	”はじめに, 対象とする複数文書, Support Vector Machine に基づく複数文書からの重要文抽出手法”	”はじめに, 対象とする複数文書, Support Vector Machine に基づく複数文書からの重要文抽出手法, 冗長性削減の有効性, まとめ”
82	日英新聞の記事および文を対応付けるための高信頼性尺度	”はじめに, 対応付けに用いた日英新聞記事, 対応付けの方針, 記事対応付けの方法, 文対応付けの方法, 記事対応付けの精度, 記事対応付けの精度向上の可能性, 文対応付けの精度, 記事対応付けの精度向上の可能性, 文対応付けの精度, 関連研究, データ公開, 今後の課題, おわりに, 付録”	”はじめに, 対応付けの方針, 文対応付けの方法, 記事対応付けの精度向上の可能性”	”はじめに, 記事対応付けの精度向上の可能性”
83	自動生成した学習データを用いた文書分類器に基づくFAQ 検索システム	”はじめに, 関連研究, 提案手法, 実験, おわりに”	”はじめに, 関連研究, 実験”	はじめに
84	Web を用いた未知語検索キーワードのソーラスノードへの割付け手法	”はじめに, 未知語分類システム, 構成技術, 未知語分類手法, 評価, 既存手法との比較, おわりに”	”はじめに, 構成技術, 既存手法との比較”	”はじめに, 未知語分類システム, 構成技術, 評価, 既存手法との比較”

次ページに続く

	タイトル	節一覧	正解データ	選択されたもの
85	コーパスに基づく動詞の多義解消	”まえがき, 関連した研究, 多義解消に必要な情報の抽出, 多義語の解消, 実験, 考察, むすび”	”まえがき, 多義解消に必要な情報の抽出, むすび”	”まえがき, むすび”

関連論文

1. Hideyuki Sawahata and Tetsuro Nishino: “Automatic extractive summarization for Japanese documents by LDA” , 11th International Congress on Advanced Applied Informatics Winter (IIAI AAI 2021-Winter), 2021.(査読あり, 本論文第 4 章対応)
2. Hideyuki Sawahata and Tetsuro Nishino: “Automatic extractive summarization for Japanese academic papers by LDA” , Information Engineering Express, Vol.9 No.2, 2023.(査読あり, 本論文第 5 章 5.1 節から 5.3 節まで対応)

参考文献

- [1] Dim En Nyaung and Thin Lai Lai Thein. Feature based summarizing from customer reviews. *International Journal of Scientific Engineering and Technology Research*, Vol. 3, No. 46, 2014.
- [2] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 11328–11339. PMLR, 13–18 Jul 2020.
- [3] 朝日新聞社. 自動要約生成 api 「tsuna」. https://cl.asahi.com/api_data/headlinegeneration.html, 2022.
- [4] NTT Communications. Ai で自然な要約文を作成する「cotoha summarize」の提供を開始. <https://www.ntt.com/about-us/press-releases/news/article/2020/0423.html>, 2020.
- [5] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 615–621, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
- [6] The Asso for Natural Language Processing. Latex corpus of the transctions of the association for natural language processing. https://www.anlp.jp/resource/journal_latex/index.html, 2020.
- [7] Alexios Gidiotis and Grigorios Tsoumakas. A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 28, pp. 3029–3040, 2020.
- [8] Makbule Ozsoy, Ilyas Cicekli, and Ferda Alpaslan. Text summarization of turkish texts

- using latent semantic analysis. In *Proceedings of the 23rd international conference on computational linguistics (Coling 2010)*, pp. 869–876, 2010.
- [9] S. Sotudeh, A. Cohan, and N. Goharian. On generating extended summaries of long documents. 2020.
- [10] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 19–25, 2001.
- [11] 西川仁, 平尾努, 牧野俊朗, 松尾義博, 松本裕治. 冗長性制約付きナップサック問題に基づく複数文書要約モデル. 自然言語処理 = Journal of natural language processing, Vol. 20, No. 4, pp. 585–612, 2013.
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [13] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM J. Res. Dev.*, Vol. 2, pp. 159–165, 1958.
- [14] Julian Kupiec, Jan O. Pedersen, and Francine R. Chen. A trainable document summarizer. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995.
- [15] S. DEERWESTER. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.*, Vol. 41, No. 6, pp. 391–407, 1990.
- [16] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, Vol. 3, No. Jan, pp. 993–1022, 2003.
- [17] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, Vol. 22, pp. 457–479, 2004.
- [18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, p. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.
- [19] Yang Liu. Fine-tune bert for extractive summarization. *CoRR*, Vol. abs/1903.10318, , 2019.
- [20] David E. Rumelhart and James L. McClelland. *Learning Internal Representations by Error Propagation*, pp. 318–362. MIT Press, 1987.
- [21] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

- [22] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks, July 2017.
- [23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, Vol. 9, No. 8, p. 1735–1780, nov 1997.
- [24] Junyoung Chung, Çağlar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv*, Vol. abs/1412.3555, , 2014.
- [25] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.
- [26] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.
- [27] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [28] 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.
- [29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, Vol. 21, No. 1, jan 2020.
- [30] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 483–498, Online, June 2021. Association for Computational Linguistics.
- [31] Alex Salcianu, Andy Golding, Anton Bakalov, Chris Alberti, Daniel Andor, David Weiss, Emily Pitler, Greg Coppola, Jason Riesa, Kuzmavn Ganche, Michel Ringgaard, Nan Hua, Ryan McDonald, Slav Petrov, Stefaen Istrate, and Terry Koo. Compact

language detector v3, 2022.

- [32] Document Understanding Conference. Duc03. <https://duc.nist.gov/duc2003/tasks.html>, 2003.
- [33] Document Understanding Conference. Duc04. <https://duc.nist.gov/duc2004/>, 2004.
- [34] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, p. 311–318, USA, 2002. Association for Computational Linguistics.
- [35] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 150–157, 2003.
- [36] C.-Y. LIN. Rouge : A package for automatic evaluation of summaries. *Proc. Workshop on Text Summarization Branches Out, Post Conference Workshop of ACL 2004*, 2004.
- [37] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72, 2005.
- [38] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.
- [39] Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020.
- [40] Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 563–578, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [41] Matan Eyal, Tal Baumel, and Michael Elhadad. Question answering as an automatic evaluation metric for news article summarization. In *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. 3938–3948, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [42] Daniel Deutsch, Tania Bedrax-Weiss, and Dan Roth. Towards question-answering as

- an automatic metric for evaluating the content quality of a summary. *Transactions of the Association for Computational Linguistics*, Vol. 9, pp. 774–789, 2021.
- [43] Ilyas Cicekli Makbule Gulcin Ozsoy, Ferda Nur Alpaslan. Text summarization using latent semantic analysis. *Journal of Information Science*, 2011.
- [44] Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 4693–4703, Online, August 2021. Association for Computational Linguistics.
- [45] 毎日新聞社. 毎日新聞 速報記事データ. <https://ndk.co.jp/newspaper-mainichi/>, 2018.
- [46] ポール・ラングマン, 今村友紀子. テンプレートでそのまま書ける科学英語論文. 羊土社, 2022.
- [47] 藤野輝雄. 理科系のためのかならず書ける英語論文. 研究社, 2008.
- [48] Xilun Chen, Ahmed Hassan Awadallah, Hany Hassan, Wei Wang, and Claire Cardie. Multi-source cross-lingual model transfer: Learning what to share. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3098–3112, Florence, Italy, July 2019. Association for Computational Linguistics.
- [49] OpenAI. Chatgpt. <https://openai.com/chatgpt>, 2023/10/8.
- [50] OpenAI. About. <https://openai.com/about>, 2023/10/8.
- [51] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, Vol. abs/2203.02155, , 2022.
- [52] 勝又智 (編). 事前学習モデルを用いた少量データに対する日本語抽象型要約, 言語処理学会年次大会発表論文集, 第 27 巻, 2021 年 3 月.
- [53] 知範小平, 守小町. Tl;dr 3 行要約に着目したニューラル文書要約. 電子情報通信学会技術研究報告 = IEICE technical report : 信学技報, Vol. 117, No. 212, pp. 193–198, 09 2017.
- [54] Muthu Kumar Chandrasekaran, Guy Feigenblat, Dayne Freitag, Tirthankar Ghosal, Eduard Hovy, Philipp Mayr, Michal Shmueli-Scheuer, and Anita de Waard. Overview of the first workshop on scholarly document processing (SDP). In *Proceedings of the First Workshop on Scholarly Document Processing*, pp. 1–6, Online, November 2020. Association for Computational Linguistics.

謝辞

はじめに、本論文において主査をつとめていただき、また多くの指導をしてくださいました電気通信大学大学院 情報理工学研究科 情報学専攻の西野哲朗教授には深く感謝申し上げます。西野哲朗教授には、研究室配属より研究のみならず、教職など様々なところで多くの指導をしてくださいました。特に研究においては多様な分野に関する知識に基づく、多角的な視点からの意見や指導が多かったように思えます。今後は教職の道に進んでまいりますが、教職においても先生のような多種多様な知識と視点を欠かさぬように精進してまいります。

また、ご多忙の中、審査委員として務めていただいた、庄野逸教授、大須賀昭彦教授、柏原昭博教授、内海彰教授にも感謝申し上げます。内海教授からは自然言語処理分野について、庄野教授からは深層学習分野についての知識と知見に基づいたご意見とご指摘を多数いただきました。大須賀教授、柏原教授からは論文に対する内容と構成について貴重なご意見や提案をいただきました。このような、ベテランとも言える教授の皆様と、直接議論できたというのは、今後の人生においても非常に貴重な経験になるかと思えます。改めてではありますが、心より感謝いたします。

そして、筆者である私の研究活動において多岐にわたる支援をしてくださいました、西野・若月研究室の皆様にも深く感謝いたします。