LLM の出力を用いた擬似ラベル学習による契約書 OCR テキストの圧縮

齋藤慎一朗 [1]　橋本航 [1]
[1]Sansan 株式会社
{shinichiro.saito, wataru.hashimoto}@sansan.com

概要

大規模言語モデル（LLM）は情報抽出に有効だが、契約書の OCR テキストは長文化しやすく、LLM の最大入力長を超えるという課題がある。既存のテキスト圧縮手法は、圧縮データの作成コストが高く、OCR テキスト特有のノイズや文の順序の乱れにより重要部分を適切に圧縮することが難しい。

本研究では、ノイズを多く含む OCR テキストを対象に、ファインチューニング済み LLM の出力に基づく擬似ラベルで学習した小型言語モデルを活用し、必要な部分のみを抽出する圧縮手法を提案する。契約書データセットでの評価により、提案手法は LLM や人手による圧縮データを用いずに、既存手法を上回る情報抽出精度を達成した。

1　はじめに

近年、大規模言語モデル（Large Language Model; LLM）の様々な自然言語処理タスクへの応用が進んでいる。特に、非構造化テキストからの情報抽出タスクにも LLM が活用されている。具体的には、非構造化テキストから構造化データを高精度に抽出するモデルを構築するために非構造化テキストと構造化されたデータのペアを大量に用意し、LLM をファインチューニングすることが効果的である。例えば、長文の契約書から契約日や契約当事者など所定の項目を抽出する場合、LLM に各項目の正解データを学習させることで、文章から所望の構造化データを取得する手法が考えられる。このように LLM のファインチューニングにより、未知の文書から所定の項目を抽出できるモデルが得られる。

一方で、契約書などの業務文書に OCR を適用して得られるテキスト (OCR テキスト) には、特有の課題が存在する。OCR テキストは文字認識の誤りやノイズを含むだけでなく、元の紙面上の配置に起因してテキストの順序が人間の読む順序と一致しない場合がある。OCR テキストからノイズを除去し、意味が通る文章を正しい順序で再構成することは容易ではなく、版面レイアウト解析 [1] や読み順推定 [2] に関する研究が進められている。また、契約書の OCR テキストは文書全体を逐語的に含むため長くなる傾向があり、LLM の最大入力長を超えてしまう場合もある。OCR テキストの長さが LLM の最大入力長を超えた場合、OCR テキストの末尾を切り捨てることで LLM への入力は可能だが、重要情報が切り捨てられることで、正しい抽出結果が得られなくなる問題が存在する。

この問題に対処するためには、OCR テキストを

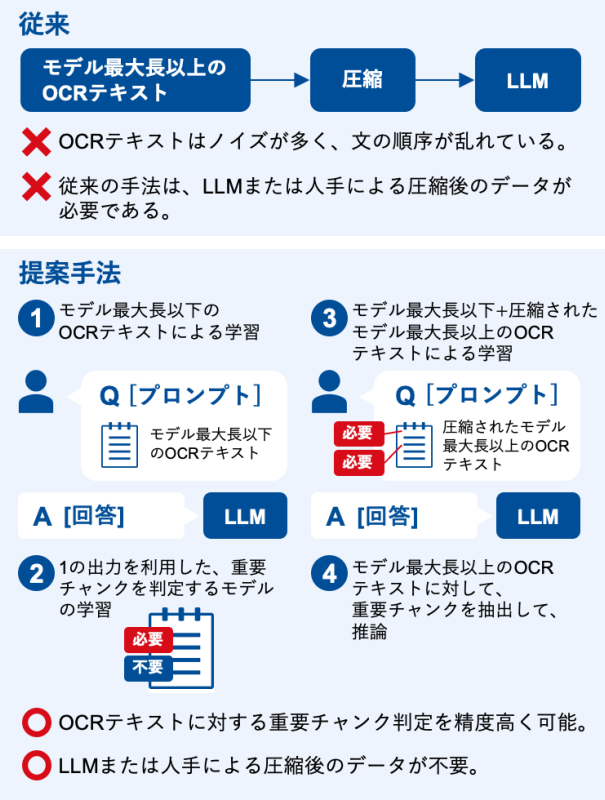

図 1　本論文の概要図

圧縮したテキスト (圧縮テキスト) を作成し、LLM の最大入力長に収めるアプローチが考えられる。既存研究では、質問に関連する情報を重視して残す圧縮手法が提案されている。具体的には、RAG（Retrieval-Augmented Generation）の検索段階で一般的に用いられるように、一定単位で区切られたテキスト (チャンク) と質問の埋め込み表現とのコサイン類似度を用いて関連度を算出し、関連性の高いチャンクのみを選択する手法が代表的である。また、LLM の入力テキストを圧縮して先頭に付与する RECOMP[3] や、テキストごとの関連度判定に基づいて圧縮を行う Provence[4] などが提案されている。

しかし、既存の圧縮手法にはいくつかの課題がある。第一に、圧縮テキスト作成に LLM あるいは人手を要する手法が多い点である。例えば RECOMP[3] では GPT-3.5 によって生成された要約テキストを学習に用いている。しかし、LLM を利用した圧縮手法では、契約書 OCR テキストのような長いテキストをそのまま入力できる LLM が限られており、利用可能な LLM が制約される。また、人手による圧縮テキストの作成にはコストがかかる。第二に、OCR テキストのようにノイズや順序乱れを含むテキストは要約自体が難しく、重要な情報を適切に保持したまま圧縮することが困難である。以上のように、人手あるいは LLM による圧縮テキストを作成せずに、ノイズや順序の乱れが存在する長文から必要情報のみを高精度に抽出・圧縮する手法が求められている。

本研究では、上記課題に対応すべく、OCR テキストのようなノイズや順序乱れが多く、かつ長いテキストに対して、擬似ラベルを用いたテキスト圧縮手法を提案する。提案手法では、新たな圧縮データを用意する必要がない。さらに、ファインチューニングされた LLM 自身の出力を用いて重要なテキストの判定を行うため、ノイズや文の順序の乱れが多い OCR テキストに対しても、情報抽出タスクを解くために必要十分な情報のみを残すと期待できる。図 1 にその概要図を示す。

2 問題設定

モデルの最大入力長よりも短いテキストデータを D_{short}、モデルの最大入力長よりも長いテキストデータを D_{long} 、D_{long} についてモデルの最大入力長より長いテキストを切り捨てたデータ D_{truncate} 、評価データを D^{*}_{long}、プロンプトを q、正解を a と定義する。圧縮器 Compressor による圧縮テキスト $D'_{\text{long}} = \text{Compressor}(D_{\text{long}})$ を作成し、q と D'_{long} を LLM に入力して推論結果 a^{*} を出力する。

3 提案手法

次の 4 段階からなる。

1. **擬似ラベル作成のための LLM のファインチューニング**：(q, D_{short}, a) と $(q, D_{\text{truncate}}, a)$ の両方を含む学習データを用いて LLM をファインチューニングする。
2. **重要チャンク判定モデルの学習**：ステップ 1 の LLM を利用し、D_{long} 内の各テキストについて「そのテキストを含めることで回答が変化するかどうか」を調べてラベルを作成する。具体的には、プロンプト q と D_{long} を入力した際に、D_{long} 内の特定の単位でのテキスト (チャンク) を含めた場合の回答と削除した場合の回答を比較し、回答内容が変わる場合はそのチャンクを「必要」、変わらない場合は「不要」と擬似ラベルをつける。これをテキスト全体について行い、得られた擬似ラベルを用いて、小型の言語モデル（例：DeBERTa-v3-base[5]）をファインチューニングする。その結果、与えられたチャンクが回答に必要か否かを判定するモデル（重要チャンク判定モデル）が得られる。
3. **重要チャンク判定モデルを用いた LLM の学習**：重要チャンク判定モデルを Compressor として用いて、D_{long} から圧縮テキスト $D'_{\text{long}} = \text{Compressor}(D_{\text{long}})$ を生成する。D'_{long} を利用し、(q, D_{short}, a) と (q, D'_{long}, a) の両方を含む学習データを用いて、ステップ 1 で学習した LLM と同じ設定でファインチューニングする。
4. **重要チャンク判定モデルと LLM を用いた推論**：重要チャンク判定モデルに D^{*}_{long} を入力し、圧縮テキスト $D^{*'}_{\text{long}}$ を得た後に、$D^{*'}_{\text{long}}$ と q を用いて、推論結果 a^{*} を生成する。

提案手法の擬似コードを付録のセクション A に載せた。

4 実験設定

4.1 モデル

情報を抽出する LLM には、最大長 8,192 トークンに対応した tokyotech-llm/Swallow-7b-instruct-v0.1[6]

モデルを用い、QLoRA[7] によるファインチューニングを、学習率は 0.1 とし、エポックは 1 で実施した。LoRA の設定は、付録のセクション B に載せた。また、ファインチューニングに利用したプロンプトを付録のセクション C に載せた。モデルは 4bit で量子化した。LLM は契約書に関連する 9 項目を出力するように学習し、項目の間は __ という文字列で区切るようにした。ここで 9 項目とは、タイトル、関係者、契約締結日、契約開始日、契約終了日、自動更新、解約通知期限、自動更新期間、金額を指す。

提案手法の圧縮器には microsoft/deberta-v3-base[5] を使用した。契約書の OCR テキストを約 512 トークン前後のチャンクに分割し、それぞれについて擬似ラベル（必要/不要）を 2 値分類で予測するようファインチューニングを行った。ここで、チャンクの長さを約 512 トークンとした理由は、使用した deberta-v3-base の最大入力長に制限があるためである。出力として確率の高いテキストのみを順序を保って連結し、8,000 トークン以下の圧縮されたテキストを生成する。ここで 8,192 トークンではなく 8,000 トークンとした理由は、後でプロンプトを追加する余白を持たせるためである。

4.2 データセット

Sansan 株式会社での契約書データ化の処理結果を統計処理しサンプリングした契約書データを用いて実験を行った。学習データとして契約書 10,000 件に対する OCR テキストと正解、評価データとして契約書 500 件に対する OCR テキストと正解を使用した。学習データ中の OCR テキスト長の内訳は、9,000 件が 8,000 トークン以下、1,000 件が 8,000 トークン超であり、評価データは全件が 8,000 トークン以上である。ここで、トークン数の計測には tokyotech-llm/Swallow-7b-instruct-v0.1[6] の語彙の辞書を利用した。

4.3 比較手法

比較手法として、次の 3 種類の手法を評価した。

1. **ベースライン**：8000 トークン以下となるよう、テキストの末尾を切り捨てる方法である。
2. **Provence**：多言語対応の公開モデルである naver/xprovence-reranker-bgem3-v1[8] をそのまま利用し、重要チャンクのみを残す方法である。モデルに対する質問は、付録のセクション D に載せた。
3. **Retrieval Augmented Generation (RAG)**：契約書テキストを 512 トークンのチャンクに分割し、質問およびチャンクを埋め込み表現に変換する。次に、質問とコサイン類似度が高いチャンクを抽出した上で、8000 トークン以下となるように順序を保ったままチャンクを選択し、LLM に入力する。埋め込みモデルには Azure OpenAI の text-embedding-3-large[9] を利用した。用いたプロンプトは、付録のセクション E に載せた。

圧縮テキストを用いて LLM をファインチューニングする際には、学習データ中の長いテキストの契約書について、それぞれの手法で圧縮したテキストに置き換えた。

4.4 評価指標

圧縮の性能を、圧縮テキストにおける正解情報の保持率によって評価した。具体的には、評価データから 10 件をランダムに抽出し、各データについて圧縮テキスト内に抽出対象である 9 項目の情報が含まれているか否かを目視で確認した。次に各圧縮手法について 9 項目の情報が圧縮テキストに残存しているかの割合を算出し、圧縮性能の指標とした。

各手法の出力の精度評価には、モデルの出力が正解と文字列完全一致した割合を示す完全一致率を利用した。9 項目に分割できない場合は、出力に失敗したと考え、全項目の出力を不正解として扱った。

各手法の推論速度として、1 データに対する秒数を計測した。計測には、AWS の g6e.xlarge の環境を利用した。GPU アーキテクチャは NVIDIA L40S であり、GPU 数は 1、GPU メモリは 48GB である。推論の際には、vLLM[10] のオフライン推論によるバッチ処理を利用した。バッチサイズは 1024 とした。

5 実験結果と考察

5.1 圧縮性能

表 1 に、圧縮後のテキストに正解情報が含まれている割合を示す。提案手法は 0.97 と最も高い保持率を示し、圧縮後も多くの必要情報を保持できていることが確認された。RAG 方式は 0.89、ベースラインは 0.80 であり、Provence 方式は 0.73 と最も低い結果となった。

Provence では正解情報を十分に保持できない傾向が確認された。Provence と提案手法の差分を分析し

表 1　圧縮テキスト中に正解情報が含まれる割合の比較

手法	正解情報の保持率
ベースライン	0.80
Provence	0.73
RAG	0.89
提案手法	**0.97**

た結果、ノイズや順序の乱れを含むテキストにおいて、Provence の圧縮がうまく機能しないケースが多く見られた。一方で、Provence では公開モデルをそのまま利用しているため、圧縮後テキストを用いて Provence のモデルをファインチューニングすることで、性能が改善する可能性が高いと考えられる。

RAG 方式と提案手法を比較すると、文脈理解を要する情報の保持に差が見られた。例えば、「契約の有効期間は契約締結日から 1 年とする」といった複数チャンクを参照して初めて特定できる記述について、RAG 方式では必要な文が選択されないケースが確認された。これは、語彙類似度に基づく retrieve では、直接的なキーワードを含まないチャンクが過小評価されるためと考えられる。一方、付録のセクション E に示すように複数項目をまとめた質問では、他項目の情報が類似度判定時のノイズとなる可能性がある。そのため、項目ごとに情報を分けて類似度判定を行う手法が有効である可能性がある。

これに対し提案手法では、LLM を用いた情報抽出における出力変化というタスクに沿った指標に基づき重要度を判定しているため、回答生成に寄与する重要なチャンクを高い精度で保持できていた。

以上より、提案手法はノイズや順序乱れを含む長い OCR テキストに対しても、必要な情報を高精度に保持しつつ圧縮できることが示された。

5.2　推論性能

表 2 に各手法の推論結果に対する正解との完全一致率を示す。先頭からの単純な切り捨て入力を行うベースラインは完全一致率 0.533 であった。Provence 方式は完全一致率は 0.383 に留まり、ベースラインより低下した。RAG 方式は 0.573 となり、ベースラインに対する改善が見られた。提案手法は 0.658 と、他の比較手法を上回る完全一致率を達成した。この結果から、提案手法により長いテキストから重要情報を保持した圧縮を用いて、高い精度による推論が可能であることが示された。

表 2　推論結果における完全一致率の比較

手法	完全一致率 ↑
ベースライン	0.533
Provence	0.383
RAG	0.573
提案手法	**0.658**

表 3　各手法の推論速度

手法	推論速度（秒 / 件）↓
ベースライン	**1.03**
Provence	2.29
RAG	3.52
提案手法	1.83

5.3　推論速度

表 3 に各手法の推論速度（1 件あたりの処理時間）を示す。ベースラインは前処理を伴わないため最も高速で、平均約 1.03 秒/件で推論が完了した。提案手法は重要チャンク抽出を含む二段階推論となるものの、小型の言語モデルによる推論が高速であり、推論時間は約 1.83 秒/件に抑えられている。

一方、Provence では約 2.3 秒/件を要し、RAG 方式では約 3.5 秒/件と処理時間が長くなった。よって、提案手法はベースラインよりは遅いものの、他の圧縮手法と比べて効率的であることが分かる。

6　おわりに

本研究では、OCR テキストのようにノイズや順序乱れを含む長文テキストから必要な情報のみを抽出・圧縮する手法を提案した。提案手法は、LLM や人手による圧縮済みデータを用意することなく、ファインチューニングした LLM を用いてチャンク単位の擬似ラベルを生成し、それに基づいて重要チャンク判定モデルを構築することで、長文入力が LLM の最大入力長を超過する問題を解決する。

契約書 OCR テキストを用いた評価では、提案手法が比較手法を上回る精度を達成した。また、提案手法はベースラインを除く比較手法よりも高速に推論可能であることがわかった。

一方で、本論文の実験設定では、比較手法を対象データに十分適応させた評価には至っていない。今後は、データ特性に特化した手法との比較検証に加え、重要チャンク判定と情報抽出を統合したモデル構成による高速化や、異なるドメインへの適用可能性の検証を通じて、提案手法の汎用化を図る。

参考文献

[1] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In **Proceedings of the 30th ACM International Conference on Multimedia**, MM '22, p. 4083–4091, New York, NY, USA, 2022. Association for Computing Machinery.

[2] Zilong Wang, Yiheng Xu, Lei Cui, Jingbo Shang, and Furu Wei. LayoutReader: Pre-training of text and layout for reading order detection. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, **Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing**, pp. 4735–4744, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[3] Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In **The Twelfth International Conference on Learning Representations**, 2024.

[4] Nadezhda Chirkova, Thibault Formal, Vassilina Nikoulina, and Stéphane CLINCHANT. Provence: efficient and robust context pruning for retrieval-augmented generation. In **The Thirteenth International Conference on Learning Representations**, 2025.

[5] Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing, 2021.

[6] Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. Continual pre-training for cross-lingual llm adaptation: Enhancing japanese language capabilities. In **Proceedings of the First Conference on Language Modeling**, COLM, University of Pennsylvania, USA, October 2024.

[7] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. **arXiv preprint arXiv:2305.14314**, 2023.

[8] NAVER. xprovence reranker (bge-m3). https://huggingface.co/naver/xprovence-reranker-bgem3-v1, 2024.

[9] Microsoft Azure OpenAI. text-embedding-3-large. https://learn.microsoft.com/azure/ai-services/openai/concepts/models#embeddings, 2024. Azure OpenAI Service embedding model.

[10] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In **Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles**, 2023.

A 提案手法の疑似コード

Algorithm 1: 提案手法のフロー

Input: LLM, Compressor, プロンプト q, 学習データ $\mathcal{T}$, 評価データ D^*_{long}

Output: 推論結果 a^*

1. 擬似ラベル作成用 LLM のファインチューニング：

$\text{LLM} \leftarrow \text{FineTune}(\text{LLM}, \{(q, D, a) \in \mathcal{T} \mid D \in D_{\text{short}} \cup D_{\text{truncate}}\})$

2. 重要チャンク判定モデル (Compressor) の学習：

$\mathcal{L} \leftarrow \emptyset$;

foreach 長文データ $D_{\text{long}} \in \mathcal{T}$ **do**

  $y \leftarrow \text{LLM}(q, D_{\text{long}})$;

  foreach チャンク $s \in \text{Chunks}(D_{\text{long}})$ **do**

    $y_s \leftarrow \text{LLM}(q, D_{\text{long}} \setminus \{s\})$;

    $l_s \leftarrow \mathbb{I}[y_s \neq y]$;

    $\mathcal{L} \leftarrow \mathcal{L} \cup \{(s, l_s)\}$;

$\text{Compressor} \leftarrow \text{FineTune}(\text{Compressor}, \mathcal{L})$

3. Compressor を用いた LLM の再学習：

$\mathcal{T}' \leftarrow \{(q, D_{\text{short}}, a)\} \cup \{(q, \text{Compressor}(D_{\text{long}}), a)\}$;

$\text{LLM} \leftarrow \text{FineTune}(\text{LLM}, \mathcal{T}')$

4. 推論：

$D^{*'}_{\text{long}} \leftarrow \text{Compressor}(D^*_{\text{long}})$;

$a^* \leftarrow \text{LLM}(q, D^{*'}_{\text{long}})$;

return a^*

B LoRA の設定

表 4 LoRA の設定

項目	設定値
r	8
target modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
LoRA α	32
LoRA dropout	0.05
Bias	none

C LLM による抽出のプロンプト

与えられた契約書の OCR テキストから、データ抽出を行なってください。

OCR テキスト:{OCR_TEXT}

図 2 LLM による抽出のプロンプト

D Provence のプロンプト

与えたテキストは契約書に対する OCR テキストです。 次の各項目について、該当する情報を抽出してください。

タイトル: 契約書のタイトルです。
関係者: 契約書における契約当事者を指します。
契約締結日: 契約が成立した日を指します。
契約開始日: 対象の契約の契約期間の開始日を指します。
契約終了日: 対象の契約の契約期間の終了日を指します。
自動更新: 双方解約の意志を示さない場合、対象の契約に自動的に契約更新がかかる仕組みがあるかどうかを指します。
解約通知期限: 自動更新が 1 の場合、いつまでに解約通知を行う必要があるかの期日の初日を表します。
自動更新期間: 自動更新が 1 の場合、自動更新の更新期間を指します。
金額: 契約当事者間で発生する・該当の契約における主対象となる金額 (税抜) を指します。

図 3 Provence のプロンプト

E RAG のプロンプト

次の要素に関連する情報のみを残してください。

タイトル: 契約書のタイトルです。
関係者: 契約書における契約当事者を指します。
契約締結日: 契約が成立した日を指します。
契約開始日: 対象の契約の契約期間の開始日を指します。
契約終了日: 対象の契約の契約期間の終了日を指します。
自動更新: 双方解約の意志を示さない場合、対象の契約に自動的に契約更新がかかる仕組みがあるかどうかを指します。
解約通知期限: いつまでに解約通知を行う必要があるかの期日の初日を表します。
自動更新期間: 自動更新の更新期間を指します。
金額: 契約当事者間で発生する・該当の契約における主対象となる金額 (税抜) を指します。

図 4 RAG のプロンプト