医薬品添付文書からの薬剤情報抽出システム

小島　諒介　岩田 浩明　中津井 雅彦　奥野　恭史
京都大学 医学研究科
{kojima.ryosuke.8e, iwata.hiroaki.3r, nakatsui.masahiko.5n, okuno.yasushi.4c}@kyoto-u.ac.jp

1　はじめに

新規薬剤の開発は非常にコストがかかるため，様々な方法を用いてそのコスト削減が取り組まれている．特に，既存の市販薬剤は上手く開発が成功した例であり，その情報を有効活用することによるより効率的な製剤開発が期待されている．近年，人工知能技術が各分野において注目されており，創薬や製薬においても市販薬剤の情報を用いての新規薬剤に関する予測モデルが多く開発されている．このような，機械学習のデータとしての用途を考えた場合，市販薬剤に関するデータを十分に収集し，十分なサンプル数を用意することが重要である．しかし，通常，市販薬剤に関する情報は各製薬企業に依る部分が大きく，製薬企業を横断的に検索可能なデータは限定された用途でしか存在せず，用途別に異なる形式で提供されているのが現状である．例えば，日本においては，薬剤の使用者にとって必要な情報は薬剤の添付文書にまとめられており，警告や使用上の注意などが記載されている．その他にも，薬剤師等の専門家向けの詳細な医薬品情報がまとめられたインタビューフォームや医薬品の承認申請のために作成する CTD（Common Technical Document）などの文書が公開されている．これらの多くは，自然言語で書かれた PDF 形式で公開されており，直接，機械学習やデータ分析可能な形式とはなっていないが，実際の製薬の現場などでは参考とされることが多く，情報資源として非常に有用であると考えられる．

これらの有用性にもかかわらず，これらの文書から情報抽出を行う研究は限定的である [1]．その大きな原因としては，これらの文書データに対するアノテーションがほとんどなされていないことが挙げられる．そのため，我々は，薬剤関連文書にアノテーションを行い薬剤情報の抽出を行うことを最終的な目標として研究を進めている．すべての文書にアノテーションを行うことは現実的に困難であるため，その一部にアノテーションを行い，残りに関しては機械学習により補完するというアプローチが有効であると考えられる．そこで，本研究ではこれらの第一ステップとして，自然言語で書かれた PDF 形式の文書から情報を抽出するためのシステムのプロトタイプを開発し，その有用性・実現可能性を評価することを目的とする．

本稿のターゲットとしてまずは添付文書を取り扱うこととする．添付文書に関しては医薬品医療機器総合機構 (Pmda) が，PDF のほかに SGML(Standard Generalized Markup Language) 形式でファイルを収集･管理しており，部分的に構造化されたデータが利用可能である．これらをマニュアルでのアノテーションの代替として利用することで，比較的容易に教師データと評価用データを用意することが可能である．

以上のことから，本研究では薬剤関連文書からの情報抽出システムを構築し，添付文書を用いて次の 4 点に考慮して評価を行う．(1) 添付文書を含む薬剤関連文書は様々な情報を含んでおり，文章以外の図や表が多く含まれる．これらの図や表を文章と分離し，個別に処理を行うためのシステム構築を行う．(2) 添付文書 SGML と添付文書 PDF から深層モデルの学習を行う．そのために，データセットの前処理を行い評価用のデータセットを構築する．(3) 実際に使う場合を想定すると，過去のアノテーションデータがどの程度未来のデータに対して汎化可能かを考慮する必要がある．本実験では 2018 年に登録されたデータで学習を行い 2019 年で新たに登録された PDF からの抽出を実行し，評価を行う．(4) 簡易的なインタフェースを作成し，データの確認を目視で容易に行えるようにし，抽出されたデータに関する議論を行う．

2　手法

図 1 はプロトタイプシステムの全体図を示す．システムは，PDF 文書からのレイアウト･文章抽出部，アノ

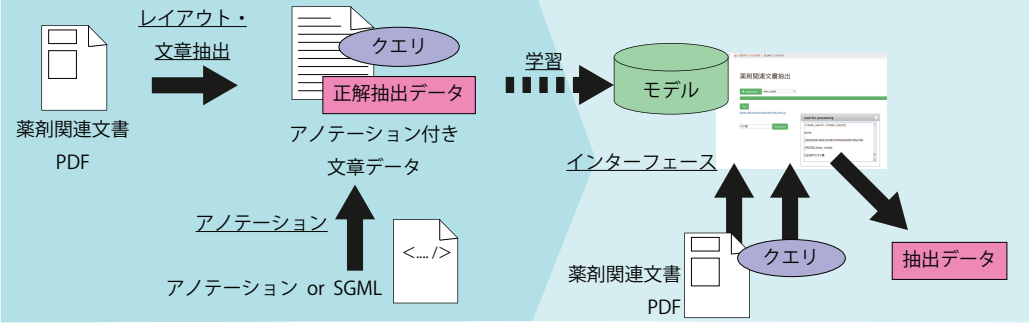

図 1: システム全体図

テーション部，情報抽出モデル学習部，インターフェースの 4 つの処理から構成される．はじめに，PDF 文書からのレイアウト・文章抽出部では PDF ファイルから本文テキストと表や図を分離し，単純なテキストを抽出する．抽出されたテキストに対してアノテーションを行う，ここでは，SGML から機械的に抽出できるものを用いて，自動的にアノテーション付けを行うこととする．情報抽出モデル学習部分は抽出されたテキストと抽出したい項目 (クエリ) と抽出されるべき部分の組からモデルの学習を行う．情報抽出部分ではクエリを受けて，そのクエリの回答となるべき部分を予測し，データ抽出を行う．

表に関しては，その認識自体が困難な問題の一つとして知られている．薬剤関連文書に関しても，線分の代わりに矩形が使われている場合や表の複数線が一つの要素で示されている場合など，見た目以上に複雑な記述方式がある．そこで，今回は PDF のファイル構造によって取得する方法のほかに，画像に変換した後，水平・垂直線を検出する手法を採用した．認識後の情報抽出に関しても，本研究では簡易的にマークダウン形式に変換し，他の文章と区別をせずに扱うこととした．また，薬剤文書には図も多く含まれており，剤型や構造式などの重要な情報も含まれるが，これらに関しても本稿の範囲外とし，これらのデータはレイアウト抽出のみ行い，情報抽出の対象からは除外した．

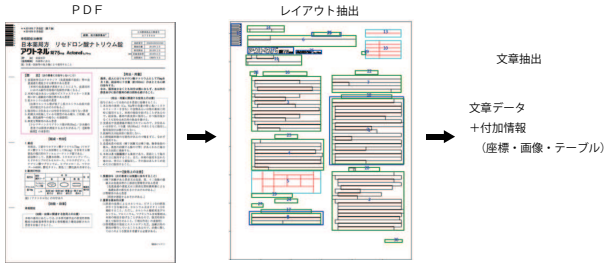

図 2: レイアウト抽出・文章抽出部分:レイアウトは，緑:テキスト, 水色:表, 青:図・矩形.

2.1 レイアウト・文章抽出部分

本システムのレイアウト・文章抽出部分は，まず，図表とテキストを分離し，それぞれに前処理・除外処理を行い，文章を抽出するために行う．図 2 に，例としてアクトネル 75mg の添付文書の 1 ページ目のレイアウト抽出結果を示す[1]．レイアウト抽出では，主に，文をブロックにまとめるという処理を行う．これは，PDF では見た目上まとまっている段落でも実際の内部構造でグルーピングされていないテキストを結合するためである．また，添付文書をはじめとした薬剤関連文書では箇条書きや見出しに重要な情報が書かれていることが多いため，文単位ではなく見出しを含めたこのブロック単位で処理することとする．

[1] https://www.info.pmda.go.jp/go/pack/3999019F1026_2_15/

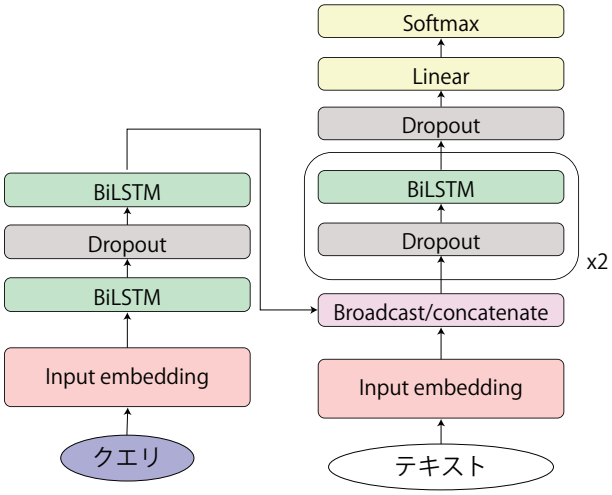

図 3: 薬剤情報抽出のための深層モデル

2.2 教師データの作成

本研究ではアノテーションの代わりに，添付文書に付随する SGML から機械的に抽出可能なラベルを用いて学習を行う．SGML から機械的なルールに従いクエリとそれに対応する正解の抽出データの対応を取り出す．より具体的には，添付文書の SGML 中の packins 直下に出現する 33 種類のタグを参考にしつつ，日本標準商品分類番号などのＩＤ関連や有効成分などの名前，有効成分の分子量や用法容量などの量に関する記述，さらに，禁忌や効能などの文章，薬物動態や臨床成績の表など多岐にわたるクエリと抽出すべきテキストのペアを抽出した．

2.3 薬剤情報抽出モデル

薬剤情報抽出モデルには深層モデルを用いた．ここでの抽出タスクはクエリとテキストの 2 つの文字列から，テキスト中のどこを抽出するかを決定するタスクである．類似タスクとして抽出型の質問回答がある．DrQA[2] や QANet[3], BERT[4] など複数のモデルが提案されているが，本研究ではプロトタイプのため図 4 に示した比較的規模の小さいネットワークを作成し，SGML によるアノテーションのみを用いて学習することとした．ネットワークは BiLSTM を含むシンプルなものを採用した．なお，出力ラベルはそれぞれ抽出箇所の先頭，内部，外部を表す B, I, O の 3 種類である．また，薬剤関連文書では未知の単語が含まれる可能性が高いため，今回のモデルでは文字埋め込みを利用している．この文字列埋め込みは Wikipedia から作成した学習済み文字ベクトルを用いた [5].

	2018 年	2019 年
PDF 数	1921	3501
利用可能な文書数	1808	2950
合計ブロック数	95535	146022
ブロックあたりの平均文字数	168.4	168.7
抽出した合計クエリ数	391930	579626

表 1: 2018 年の登録文書と 2019 年登録文書

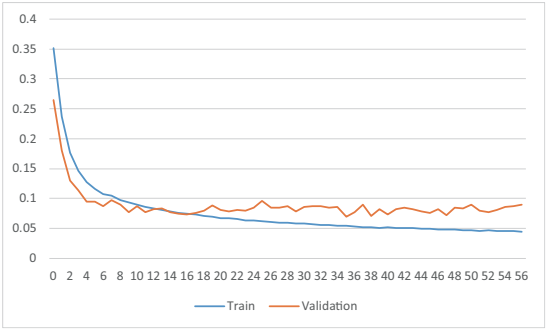

図 4: 学習中の損失 (縦軸)　vs epoch 数（横軸）

3 実験

本システムの評価を行うために，2018 年と 2019 年に登録された添付文書データを用意した．評価では，2018 年分を学習データとして利用し，2019 年分をテストデータとして評価に用いた．添付文書は改訂される場合があり，同一薬剤に対して登録には重複があるため，評価用の 2019 年分からは 2018 年分に含まれる薬剤の添付文書は取り除いて評価を行った．表 1 に実験に用いたデータの詳細を示した．今回，画像 PDF は OCR と組み合わせることで対応可能であるが，OCR のエラーと分離するために本稿では除外した．また，実際のデータではいくつかの文書において，埋め込みフォントや保護・暗号化などの理由により処理できない PDF が存在したため除外した．さたに，SGML と対応が取れなかったデータについても除外した．

学習に関しては，メモリや学習速度と性能を考慮してチューニングを行い表 2 のパラメータを選択した．また，図 4 に学習中のコスト関数の値を示した．この結果から，validation の損失が最も小さかった epoch=46 のモデルを用いて，validation データに対する結果と

項目	値
最適化	Adam
文字埋め込みの次元数	128
クエリー隠れ層次元数	512
ブロックテキスト隠れ層次元数	768
ミニバッチサイズ	128
dropout rate	0.3

表 2: モデルパラメータ

	validation (2018)	test (2019)
正答率	0.97	0.96
精度	0.75	0.57
再現率	0.86	0.65
F 値	0.80	0.61
抽出テキストの正答率	0.82	0.56

表 3: 結果：正答率・精度・再現度に関しては文字ごとの評価，抽出テキストの正答率は正解テキストと厳密一致した場合を正解とした評価

テストデータである 2019 年の添付文書に適用した結果を表 3 に示した．

この結果から，特に，登録された期間が異なる文書では F 値で 0.19 ポイント，特に厳密一致の正答率は 0.26 ポイントと大きく低下することが分かった．定性的な目視による結果の評価としては，分子式や分子量，承認番号など平文や単純な評価ら数値や記号を抜き取るクエリーには概ね対応できていたが，厳密一致は難しく単位の欠損や周囲の余計な単語を含んだフレーズを抽出することが多くあった．また，表に関しては，表全体や行全体を抽出するクエリーには比較的正解できていたが，多くの数値が並んだ表の特定のセルを抽出する必要があるクエリに関しては不正解が多く，課題が残る結果となった．

我々はこの学習済みモデルを用いたプロトタイプシステムとして，web ブラウザから API としてクエリを送信するシステムを構築した．その場合，計算時間として，クエリからの情報抽出に数秒程度の時間を要するため，インタラクティブなシステムとするためには，予め代表的なクエリに関してキャッシュをしておくなどのシステム上での工夫が必要であることも分かった．

4 おわりに

本稿では PDF で提供される薬剤関連文書のひとつである医薬品添付文書からの薬剤情報抽出システムを構築した．特に，情報抽出のために深層学習モデルを構築し，SGML から自動的にラベル付けされたデータセットを用いて，モデルの評価を行った．その結果，単純な数値や記号を抽出するタスクにおいてはある程度の正答率を達成するものの，複雑な表などにおいては低い正答率となった．

今後はマニュアルのアノテーションやキュレーションを行うことで，より高品質なアノテーションと本システムを合わせて，精度向上をすることが考えられる．また，手法の改善と合わせて，抽出されたデータを学習データに用いた薬剤に関する機械学習モデル構築といった活用も考えている．

謝辞

本研究を進めるにあたり貴重なデータを提供していただいた医薬品医療機器総合機構 (Pmda) と医薬基盤・健康・栄養研究所に感謝する．本研究は国立研究開発法人新エネルギー・産業技術総合開発機構 (NEDO) と日本医療研究開発機構 AMED の課題番号 JP19nk0101101h「製薬・医療関係文書からの情報抽出と基礎データベースの拡充」の支援を受けた．

参考文献

[1] 生駒 卓志, 津田 和彦, “テキストマイニングを利用した市販薬含有成分の効果検証”, 人工知能学会第 27 回全国大会, 2013.

[2] Chen, Danqi, et al. “Reading Wikipedia to Answer Open-Domain Questions.” In Proc. of ACL 2017 (Volume 1: Long Papers). 2017.

[3] Yu, Adams Wei, et al. “Qanet: Combining local convolution with global self-attention for reading comprehension.” arXiv preprint arXiv:1804.09541 (2018).

[4] Devlin, Jacob, et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” In Proc. of NAACL-HLT 2019, Volume 1. 2019.

[5] jaembed, `https://github.com/iki-taichi/jaembed`