1　メタデータ・画像抽出部

この章では、PDF 形式の論文からメタデータと画像を取得するフェーズについて説明する。今回抽出するメタデータは、タイトル、著者名、参考文献とする。PyMuPDF という Python のライブラリを用いて、PDF ドキュメントからテキストや画像を抽出した。PyMuPDF は、オープンソースのライブラリであり、PDF ドキュメントのデータ抽出や分析・変換等の操作や処理を行う機能を提供する。

1.1　タイトル・参考文献の取得と課題

ずタイトルを抽出するフェーズについて説明する。一般的な学術論文では、1 ページ目にその論文のタイトルが 1 番大きいフォントで必ず明記される。この規則を利用して、PDF ドキュメントの 1 ページ目のテキストデータの中から、フォントサイズが最大の文字列をタイトルとみなし、抽出した。

参考文献を抽出するフェーズについて説明する。参考文献は"[number] " という形式で、参考にした日付を記述することが多い。また、 1 つの参考文献は 5 行以内で記述されることが多い。これらの規則性を利用し、"[number] " という文字列を含む行を正規表現で取得する。これだけではノイズとして本文中で参考文献を参照した部分が含まれてしまうため、西暦年を用いてノイズを減らした。5 行以内に次に"[number]" がマッチするものがあれば、次にマッチした行の 1 行前までを 1 つの参考文献とし、5 行以内に次にマッチするものが見つからなければ参考文献ではないとした。しかし、この方法ではまだノイズが多いため、西暦年を用いてノイズを減らした。西暦年の下限は論文の発行年数が分からなかったため 1950 で、上限は 2050 で仮置きした。文字列内の数字を全て取り出し、その中に設定した西暦年の範囲内かどうかを確認する。正規表現にマッチした行から 5 行以内に西暦年が存在すれば、"[number] " がマッチした行から西暦年が存在する行までを 1 つの参考文献とし、抽出を行った。西暦年を用いた方法では、大幅にノイズを減らすことが出来た。

1.2　著者名の取得と課題

今後は著者名の取得が課題である。著者名は正規表現で取ってくることが難しい。また、研究会などで提出する論文はタイトルの下に著者名が記述される。しかし学内の卒論では、タイトル直下に著者名は記述されていないため、全ての論文で汎用的に著者名を取得するのは難しい。よって、私たちは、学内の論文と研究会などの論文はページ数が異なるという点に着目し、学内の論文と研究会の論文で分けて処理を行う予定である。

1.3　画像の取得と課題

画像の取得するも PyMuPDF の exract_image を利用して PDF ドキュメントに埋め込まれたグラフや表などの画像を抽出した。現状では、余分な画像も多く抽出されてしまうため、必要な画像を選別する必要がある。その方法として、いかの 2 つを考案した。1 つ目は、teedy にアップロードする前に不必要な画像を削除する方法である。teedy にアップロードする前に、タイトルや参考文献などのメタデータと合わせて取得した画像全てをユーザに返し、必要なものだけをユーザに選択してもらう方法である。2 つ目は、余分な画像を含む全ての画像を一度 teedy にアップロードし、ユーザが必要に応じて、その都度削除する方法である。今回は後者の全画像を teedy にアップロードする方法で実装した。

2　考察

参考文献の取得については、西暦年を活用することで大幅にノイズを減らすことができた。著者名の取得については、著者名は正規表現で取ってくることが難しく、論文の形式によって記述箇所が異なるため、全ての論文で汎用的に著者名を抽出することが難しかった。しかし、学内の論文と研究会などの論文で著者名の記述箇所に規則性が見られたため、ページ数によって学内の論文と研究会の論文で分けて処理を行うことで、対策しようと考えている。画像の取得についても、著者名と同じく、画像が抽出できない論文の形式が存在する。現在は学内の論文である卒論形式の PDF ドキュメントからは、画像を抽出することができているため、学外形式の論文からも画像を抽出できるような考案する必要がある。