センター試験英語入試問題の自動XML化について

磯崎 秀樹　佐藤 文香　木野内 友梨
岡山県立大学
isozaki@cse.oka-pu.ac.jp

要旨

国立情報学研究所の人工知能プロジェクト「ロボットは東大に入れるか」では、センター試験を計算機に自動で解かせる取り組みを行っているが、現在の設定では、試験問題が計算機にわかるように、あらかじめ人手でXML化している。このため、XML化に時間と人件費がかかる。また、インターネットでは、「人手が介在するのはずるい」などの批判がある。そこで本稿では、英語入試問題を自動でXML化する試みについて報告する。

1　はじめに

国立情報学研究所の人工知能プロジェクト「ロボットは東大に入れるか」では、センター試験の問題を計算機に自動で解かせる取り組みを行っており、岡山県立大学でも、英語の問題を解く研究を行っている。

しかし、実際にはロボットは存在せず、試験問題が計算機にわかるように、あらかじめ人手で図1に示すようにXML化しておき、それを問題を解くソフト（ソルバー）が読んで解いている。（行末の \ は、OCRの結果ではなく、長過ぎる1行をこの原稿の幅に収めるために改行したことを示す。）このため、XML化に時間と人件費がかかる。また、インターネットでは、「人手が介在するのはずるい」などの批判がある。

そこで本稿では、試験問題の画像ファイルをOCRで文字認識し、その結果を自動でXMLに変換することにより、XMLの時間と人件費を削減し、「ずるい」という批判をなくすことを目指す。

なお、自動で文書をXMLに変換するツールはいくつも存在する[1][5, 1]が、ツールによって生成されるXMLが異なる。たとえば、`pdftohtml`[2]というツールに`-xml`というオプションを付けると、PDFファイルから文字列を取り出して、それぞれの文字列の左上の座標・幅・高さ・フォントとともに出力してくれる。しかし、このXMLは、英語の問題を解く、という課題からすると低レベルすぎる。

すでに英語のソルバー[2, 4]が現在の東ロボのXMLの形式を想定して作成されており、これらのソルバーを変更するのは大変なため、現在の東ロボのXML形式を再現できるプログラムを作成する。

英語のセンター試験の問題の画像をOCRソフトで読むと、図2に示すように、センター試験に特有な、以下の場所で読み間違いが多発する。

- [8]などの四角い枠が、`"｜8｜"`や`"I8]"`などに誤読される。
- 選択肢を表す ① や ④ などの丸数字が、`@`や`ⓒ`などに誤読される。
- 「問」が`PpI`などに誤読される。

これらの読み間違いは、決まった読み間違い方をするのではなく、場合に応じて様々に読み間違えられる。

そこで、これらの読み誤りを削減するため、試験問題の画像のこれらの場所を、あらかじめ誤読しにくい文字の画像に書き換えておく。つまり、このXML化を以下の3つのステップに分ける。

(1) OCRで誤認識されやすい部分をあらかじめ画像レベルで書き換えておき、誤認識を減らすステップ。これにはOpenCV 2.4.9[3]を用いた。

(2) OCRソフトで文字認識するステップ。OCRソフトとしては、オープンソースのTesseract[4]の他、市販ソフト数種類を試したところ、ABBY FineReaderの精度が比較的良好だったので、これを用いた。

(3) 文字認識の結果得られたプレインテキストを読みこんで、人手で作成されたのと同じ形式のXMLに変換するステップ。このため、ルールベースの書き換えプログラムをPythonで作成した。

2　手法

[1] `http://www.maplesoft.co.jp/products/t2x.html`
[2] `http://pdftohtml.sourceforge.net`
[3] Version 3が公開されているが、第一著者がversion 2に慣れているので、version 2を採用した。
[4] `https://github.com/tesseract-ocr`

```
<question id="Q11" minimal="no">
 <label>【２】</label>
 <instruction>
     次の問い（Ａ～Ｃ）に答えよ。
 </instruction>
 <info>（配点　41）</info>
 <br/>
 <question id="Q12" minimal="no">
  <label>Ａ</label>
  <instruction>
      次の問い（問１～10）の<ref target="A8">8</ref>\
～<ref target="A17">17</ref>に入れるのに最も適当な\
ものを，それぞれ下の①～④のうちから一つずつ選べ。
  </instruction>
  <br/>
  <question id="Q13" minimal="yes" answer_style=\
"multipleChoice" answer_type="sentence" \
knowledge_type="IDM" anscol="A8">
    <label>問１</label>
    <data id="D1" type="text">
        I understand
        <ansColumn id="A8">
            8
        </ansColumn>
        of our students are working part-time in \
the evening to pay their school expenses.
    </data>
    <br/>
    <choices anscol="A8">
        <choice ansnum="1">
            <cNum>①</cNum>almost
        </choice>
        <choice ansnum="2">
            <cNum>②</cNum>any
        </choice>
        <choice ansnum="3">
            <cNum>③</cNum>anyone
        </choice>
        <choice ansnum="4">
            <cNum>④</cNum>most
        </choice>
    </choices>
    <br/>
  </question>
```

図 1: 東ロボで用いられるセンター試験の XML ファイルの一部（平成 25 年の試験問題の第 2 問冒頭）

2.1　前処理による文字認識誤りの削減

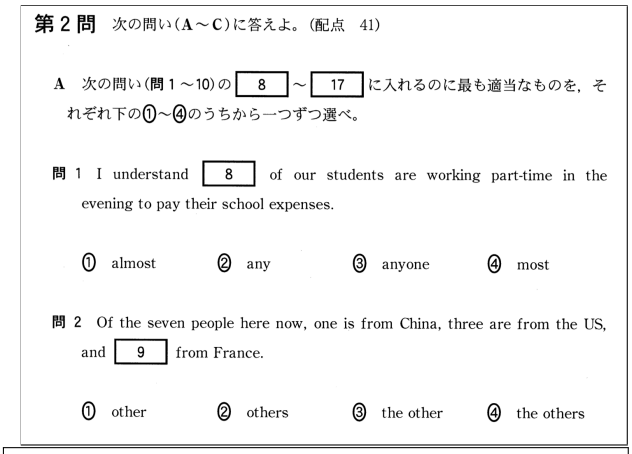
今回は、特に誤読の多いところに着目して、以下の書き換えを行った。

- ③ などの丸数字をテンプレートマッチングによって見つけ、白い矩形で塗りつぶす。さらに、あらかじめ用意した (3) のような括弧数字の画像で上書きする。

  なお、年によって試験問題画像の縦横のサイズが違うことがある。そこで、ページの縦横のサイズを調べ、それに合わせてテンプレートのサイズを変更している。

- [8] などの矩形の枠のある数字は [8] と書き換える。輪郭線を列挙して、そのうち、矩形であるものを見つけて、中に書かれている文字を tesseract で読んで、数字だけであれば、矩形の枠を白で上書きして消す。それから、左右に [と] を描画する。

- 同様に、「**第 1 問**」などの大問は「Q1Q」などの、「**問 1**」などの小問は「t 1 t」などの画像で置き換える。




```
第 2 PpI 次の問い（A～C）に答えよ。（配点 41）
A 次の問い（問 1～10）の｜8｜～｜17｜に入れるのに最も適当\
なものを，そ れぞれ下の©～のうちから一つずつ選べ。
問 1 I understand 8 oi our students are working part-\
time in tne evening to pay their school expenses.
① almost② any@ anyone@ most
問 2 Of the seven people here now, one is from L,hma,\
three are from the US, and 9 from France.
© other② others@ the other @ the others
```

図 2: 平成 25 年の試験問題第 2 問冒頭を直接 OCR で読み取った結果

丸数字の置き換えを行い、意図した通りに置き換えられたかどうかを調べたところ、平成 25～27 年の 3 年分の本試験と再試験の問題で、丸数字の総数は 669、テンプレートマッチングで見つけた数が 665 であり、正しく見つけて置き換えられたものは 633 であったので、適合率 P=95.2%、再現率 R=94.6%、F=94.9%である。

画像を書き換えた場合と直接読んだ結果の読み誤りを比較しようとしたが、直接読んだ場合の OCR 出力の中には、画像上のどの丸数字と対応しているのかすらわからないほど語順が崩れているものがあったため、本稿の執筆時点では、まだこの比較はできていない。

ただし、テンプレートマッチングのときに、マッチングのスコアがどれくらいであればマッチしたとみなすかの閾値が問題である。厳しくすると recall が下がり、緩くすると precision が下がる。予備実験の結果により、マッチングのスコアが最大値のピクセルの 0.95 倍以上のピクセルをマッチしたと見なした。

2.2 プレインテキストから XML への変換

図 3 が、前処理をした結果を ABBY FineReader で読み取った結果である。

図 1 を目標として、Python でルールを書き換えた結果が

以下のように、書き換えはタグごとに、特徴的な文字列を見つけて書き換えるプログラムを作成し、パイプでつなぐことにした。

- 大問・中問・小問の見出しを囲む<label>, </label> を入れる label.py。
- 各選択肢を囲む<choice>, </choice> を入れる choice.py。
- ある問題の全選択肢を囲む<choices>, </choices> を入れる choices.py。
- 問題に対する日本語の指示などを囲む <instruction>, </instruction> を入れる instruction.py。
- 大問・中問・小問それぞれ囲む<question>, </question> を入れる question.py。

まず、label.py は、以下の手がかりを見つけて、<label> </label>というタグを入れる。

- 小問の手がかりは「問 1」などの画像を「t 1 t」などの画像に書き換えた結果を読み取って得られる「t\s*\d+\s*t」というパターン。
- 中問（A, B, C）の手がかりは「[A-C]\s*次の」というパターン。
- 大問の手がかりは「第 1 問」などの画像を「Q 1 Q」などの画像に書き換えた結果を読み取って得られる「Q\s*\d+\s*Q」というパターン。

次の choice.py では、①, ② などの画像を (1), (2) などに書き換えて得られる「\((\d+)\)」というパターンを見つけて、各選択肢を囲む<choice> <cNUm> </cNum> </choice>を以下のように入れる。

```
<choice ansnum="1">(1)<cNum>(1)</cNum>
generate </choice>
<choice ansnum="2">(2)<cNum>(2)</cNum>
enius </choice>
```

また、これとは別に 3 などの枠が前処理で書き換わった [3] を「\[\s*(\d+)\s*\]」で見つけて、書き換える。<ansColumn id="A3">3</ansColumn> というタグに書き換える。

```
Q2Q 次の問い（A〜C）に答えよ。（配点 41）
A 次の問い（問 1〜10）の [8 ]〜[17 ] に入れるのに最も適当な\
ものを，そ れぞれ下の (1)〜(4) のうちから一つずつ選べ。
t 1 t ⊥  understand [8] of our students are \
working part-time in the evening to pay their \
school expenses.
(1)almost(2)any(3)anyone(4)most
t 2 t Of the seven people here now, one is from \
China, three are from the US, and [9 ] from France.
(1)other(2)others(3)the other(4)the others
```

図 3: 前処理をしてから OCR で読み取った結果

この書き換えでは、「以下の [8]〜[13] に入れるのに」のような表現にマッチしないように、\]\s*〜\s*\[とはマッチしないことを確認する。

次に choices.py によって選択肢全体を囲む <choices anscol="A3"> などのタグを入れる。anscol の数字は、直前に現れる 3 の枠の中の数である。これは choice.py が入れた <ansColumn id="A3">3 </ansColumn> を見つけて、この情報を覚えておく。

ここまでの処理をした結果が図 4 である。

instruction.py では、</label> が<instruction> の手がかりとなる。</instructio>の手がかりになるのは、行末の「。」であるが、しばしば「(配点」が後続することがあるので、「。(配点」にも対応するようにルールを書く。

最後に question.py によって、大問・中問・小問をそれぞれ囲む<question>タグを入れる。label.py が出力した以下の手がかりが利用できる。

- 大問：<label>【(\d+)】</label>
- 中問：<label>([A-C])</label>
- 小問：<label>問 (\d+)</label>

</question> は、<question> を出力する直前に出力する。

- 新しい大問が始まるとき、直前の大問・中問・小問を閉じる。
- 新しい中問が始まるとき、直前の中問・小問を閉じる。
- 新しい小問が始まる時、直前の小問を閉じる。
- OCR が出力したファイルの最後では、新しい大問が始まるときと同じ処理をする。

なお、表の XML 化については表が読めれば [3] 難しくない。

```
<label>【2】</label>次の問い（A〜C）に答えよ。（配点 41）
<label>A</label>次の問い（問 1〜10）の [8 ]〜[17 ] に入れるのに\
最も適当なものを，そ れぞれ下の (1)〜(4) のうちから一つずつ選べ。
<label>問 1</label> ⊥ understand
<ansColumn id="A8">8</ansColumn> of our students are \
working part-time in the
evening to pay their school expenses.
<choices anscol="A8">
<choice ansnum="1">(1)<cNum>(1)</cNum>almost</choice>
<choice ansnum="2">(2)<cNum>(2)</cNum>any</choice>
<choice ansnum="3">(3)<cNum>(3)</cNum>anyone</choice>
<choice ansnum="4">(4)<cNum>(4)</cNum>most</choice>
</choices>
<label>問 2</label> Of the seven people here now, one is \
from China, three are from the US, and
<ansColumn id="A9">9</ansColumn> from France.
<choices anscol="A9">
<choice ansnum="1">(1)<cNum>(1)</cNum>other</choice>
<choice ansnum="2">(2)<cNum>(2)</cNum>others</choice>
<choice ansnum="3">(3)<cNum>(3)</cNum>the other</choice>
<choice ansnum="4">(4)<cNum>(4)</cNum>the others</choice>
</choices>
```

図 4: OCR 結果に XML のタグを自動で入れている途中の状態

3 まとめ

『ロボットは東大に入れるか』プロジェクトでは、センター試験の問題を人手で XML に変換している。これには時間や人権費がかかり、また人手が介在することへの批判もある。そこで本稿では、自動 XML 化を試みた。市販 OCR ソフトを試したところ、②や 3 などのセンター試験特有の表現で誤読が頻発したため、試験問題の画像をあらかじめ前処理で書き換えておくことで誤読を減らすことができた。また、OCR ソフトの出力するプレイン・テキストに現在東ロボで使用されている XML タグを入れるための手がかりを明らかにした。

なお、機械学習を利用した XML 化も考えられるが、今回は XML 化された試験問題が少ないことなどの理由により、ルールベースを用いた。

なお、現在のプログラムでも、市販 OCR ソフトとのデータのやりとりは人手で行っている。Tesseract の認識率が向上すれば、ここの部分は Tesseract で置き換えられるので、自動化できるはずである。

謝辞

本研究は NTT との共同研究である。本研究を推進するにあたって、大学入試センター試験問題のデータをご提供くださった独立行政法人大学入試センター及び株式会社ジェイシー教育研究所に感謝します。

参考文献

[1] Hervé Déjean and Jean-Luc Meunier. A system for converting PDF documents into structured XML format. In *IAPR International Workshop on Document Analysis Systems*, pp. 129–140, 2006.

[2] 東中竜一郎, 杉山弘晃, 磯崎秀樹, 菊井玄一郎, 堂坂浩二, 平博順, 南泰浩. センター試験における英語問題の回答手法. In *Proc. of the Annual Meeting of the Association for Natural Language Processing*, pp. 187–190, 2015.

[3] 磯崎秀樹, 伊藤圭汰, 荒木良元. 論文 QA のための画像処理 表を読む . In *Proc. of the Annual Meeting of the Association for Natural Language Processing*, pp. 139–142, 2015.

[4] 松崎拓也, 横野光, 宮尾祐介, 川添愛, 狩野芳伸, 加納隼人, 佐藤理史, 東中竜一郎, 杉山弘晃, 磯崎秀樹, 菊井玄一郎, 堂坂浩二, 平博順, 南泰浩. 「ロボットは東大に入れるか」プロジェク代ゼミセンター模試タスクにおけるエラーの分析. In *Proc. of the Annual Meeting of the Association for Natural Language Processing*, 2015.

[5] 渡邉茂樹, 布目光生, 後藤和之. ビジネス文書の自動 Xml 化が拓く新たな応用領域.「東芝ソリューション テクニカルニュース」 2008 年夏季号 Vol.14, pp. 12–13, 2008.