論文QAのための画像処理 ～表を読む～

磯崎 秀樹　伊藤 圭汰　荒木 良元
岡山県立大学
isozaki@cse.oka-pu.ac.jp

要旨

論文・白書・新聞など、多くの文書には、表やグラフがある。しかし、著者の知る限り、これまでの自然言語処理は、文章だけを扱い、表やグラフは完全に無視してきた。我々は、自然言語処理分野の英語論文を対象とした質問応答システム（**論文 QA**）を作成しているが、表やグラフを読めば簡単に答えられる質問に答えることができない。そこで、表やグラフを読むツールの作成を始めた。本稿では、グラフより簡単そうな表を読むのも意外に難しいことを指摘し、現在の解読アルゴリズムについて述べる。

1　はじめに

近年、世界中で英語論文が多数出版されている。しかし、日本語と英語は語順などがあまりに違うため、日本人は一般的に英語が苦手であり、さらに、専門用語や難解な数学だらけの最先端の英語論文を読むのには多大な苦労が伴う。

そこで我々は、自然言語処理分野の英語論文を対象として、ACL Anthology[1] を中心とした論文約 2 万本を解析し、検索できるようにしており [4]、現在はこれを進めて、以下のような質問に答える質問応答システム（以下、**論文 QA** と呼ぶ）を作ろうとしている。

- C-value の定義は？
- （NLP で）線形計画法を最初に使ったのは誰？
- 英語の品詞タグ付けの精度は今どれくらい？
- 翻訳自動評価にはどんな方法がある？
- Schank のスクリプトに触れている最近の論文は？

ベースとなる英語の論文 QA システムをまず作り、日本語の質問を解析して英語の論文 QA システムを呼び出すモジュールを追加する [6]。

検索に基づく質問応答システム [5] の研究は 10 年以上前には盛んで、最近でも、ドコモの「しゃべってコンシェル [1]」や IBM の「ワトソン [9]」のように一般の人にも見える成果は出ているものの、答の信頼性は未だに低い。「オープンドメイン」を標榜していたが、ドメインの広さと信頼性はトレードオフの関係にあり、オープンドメインである限り、高い信頼性を保証するのは難しい。

我々は自然言語処理分野に限定することで信頼性を上げやすくする。ドメインを限定しているとはいえ、論文の数は多く、検索によるアプローチは有効である。また、論文によって実験条件が違ったり、年により成績が上がったりするので、これらを分かりやすく表示してくれるシステムがあれば助かるであろう。

論文には多数の表・グラフが含まれており、筆者の主張を裏付ける重要な役割を果たしているが、自然言語処理で対処できないので、これまで処理の対象とされてこなかった。

しかし、図表の領域を検出したり [3]、定型的な表形式帳票を自動処理する研究 [8]、表から罫線抽出を行い、領域分けする研究 [10] などがすでにある。

そこで、質問応答システムという観点から、表やグラフを処理しやすいテキストに変換するツールを作成することにした。市販 OCR ソフトの中には表を読んで Excel などに変換できるものもあるが、論文 QA システムに組み込みたいので、市販ソフトではなく、オープンソース・ソフトをベースにして作ることにした。具体的には、オープンソースの画像認識ソフト OpenCV[2] と文字認識ソフト Tesseract OCR[3] を用いた。

なお、文字認識は画像の解像度やノイズ、傾きなどの影響を大きく受けるが、PDF から OCR 用の画像を作成する場合、ノイズや傾きの問題はほとんどない。大量の PDF ファイルを処理する場合は、バッチ処理により、直接画像フォーマットに変換できる。ACL Anthology では、2000 年より以前の古い論文がスキャンされているが、ここでは考えない。

以下では、表を読むアルゴリズムを説明する。

[1] http://www.aclweb.org/anthology/

[2] http://opencv.org/

[3] https://code.google.com/p/tesseract-ocr/

入力と出力

(a)

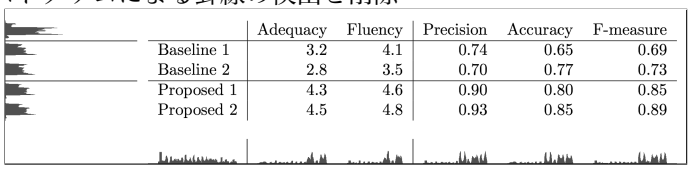
	Adequacy	Fluency	Precision	Accuracy	F-measure
Baseline 1	3.2	4.1	0.74	0.65	0.69
Baseline 2	2.8	3.5	0.70	0.77	0.73
Proposed 1	4.3	4.6	0.90	0.80	0.85
Proposed 2	4.5	4.8	0.93	0.85	0.89

(b)

```
----          ----          ----
     :
Baseline_1    Adequacy      3.2
Baseline_1    Fluency       4.1
Baseline_1    Precision     0.74
     :
Proposed_2    Adequacy      4.5
Proposed_2    Fluency       4.8
Proposed_2    Precision     0.93
     :
```

ヒストグラムによる罫線の検出と削除

(c)



(d)

	Adequacy	Fluency	Precision	Accuracy	F-measure
Baseline 1	3.2	4.1	0.74	0.65	0.69
Baseline 2	2.8	3.5	0.70	0.77	0.73
Proposed 1	4.3	4.6	0.90	0.80	0.85
Proposed 2	4.5	4.8	0.93	0.85	0.89

太文字化後のヒストグラムによる行・列の座標の決定

(e)

	Adequacy	**Fluency**	**Precision**	**Accuracy**	**F-measure**
Baseline 1	**3.2**	**4.1**	**0.74**	**0.65**	**0.69**
Baseline 2	**2.8**	**3.5**	**0.70**	**0.77**	**0.73**
Proposed 1	**4.3**	**4.6**	**0.90**	**0.80**	**0.85**
Proposed 2	**4.5**	**4.8**	**0.93**	**0.85**	**0.89**

(f)

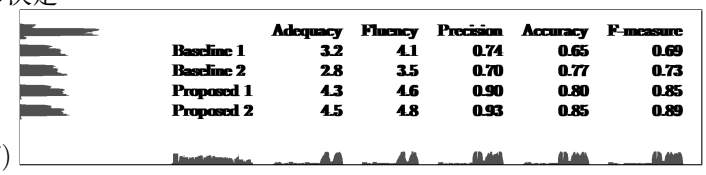


図 1: 表画像をセルに分割する過程

2 手法

2.1 単純な表のテキスト化

たとえば図 1(a) のような表を読み取って QA で使うのに扱いやすいデータベース、あるいは「Baseline 1 の Adequacy は 3.2 である。」のような文章に変換したい。(b) の形式ができれば、これをデータベースに格納したり、「Baseline 1 の Adequacy は 3.2 である。」として出力するのは簡単である。

当初、罫線を頼りにして、罫線で囲まれた矩形を抽出して文字を読む方法を試したが、英語圏では罫線の使用が推奨されておらず [4]、実際に罫線の少ない表が多いので、罫線はあらかじめ消してから解読する。

まず、各 X 座標、Y 座標ごとに黒いピクセルの数を数えて (c) の左や下のようなヒストグラムを作る。左の山は、各 Y 座標ごとに、黒ピクセルの数を横方向に数えたものである。下の山は各 X 座標ごとに、黒ピクセルの数を縦方向に数えたものである。どちらも、罫線のところに細くて強いピークがある。このヒストグラムから罫線の位置を求め、罫線を削除して (d) のような罫線のない表にする。

次にこの画像を左右にずらして重ねて (e) のように太くする。これは、単語の途中のわずかな空白などを列の切れ目と認識されないようにするためである。この結果に対して再度ヒストグラムを計算して (f) を得る。すると、行や列の境界がヒストグラムの谷としてわかるので、谷を境界として、各セルの四隅の座標を決定する。この座標を用いて (d) をセルに分割する。

こうして得られた各セルの画像を Tesseract で読むことで、各セルの中に何と書かれているのかわかる。そして、一番左を行ラベル、一番上を列ラベルとして、各セルの値を [行ラベル　列ラベル　値] という三つ組の形式で出力すれば、自然言語処理やデータベースで扱いやすい (b) が得られる。

(b) は 5 行× 6 列の各セルにつき、1 行ずつ出力されるので、30 行出力されている。空のセルは、当然 OCR の出力がないが、それでは扱いづらいので、OCR の出力がないことを “____” という文字列で表している。1 行 1 列のセルは空なので、行ラベルも列ラベルも空になり、“____　____　____” と出力されている。

2.2 市販 OCR ソフトの性能

市販の日本製 OCR ソフトで Excel に変換できるもので、同じ表を解読して作成した Excel ファイルを図 2 (a) (b) に示す。

どちらも行がずれており、また、罫線を頼りにセル

[4] たとえば、LaTeX の英語のマニュアル [7] には、The tabular environment makes it easy to draw tables with horizontal and vertical lines. Don’t abuse this feature. Lines usually add clutter to a table: ... と書かれており、罫線の濫用を戒めている。

(a)

	Adequacy	Fluency	Precision	Accuracy	F-measure
Baseline 1	3.2	4.1	0.74	0.65	0.69
Baseline 2	2.8	3.5	0.70	0.77	0.73
Proposed 1	4.3	4.6	0.90	0.80	0.85
Proposed 2	4.5	4.8	0.93	0.85	0.89

(b)

	Adequacy	Fluency	Precision	Accuracy	F-measure
Baseline 1	3.2	4.1	0.74	0.65	0.69
Baseline 2	2.8	3.5	0.70	0.77	0.73
Proposed 1	4.3	4.6	0.90	0.80	0.85
Proposed 2	4.5	4.8	0.93	0.85	0.89

(c)

	A	B	C	D	E	F
1		Adequacy	Fluency	Precision	Accuracy	F-measure
2	Baseline 1	3.2	4.1	0.74	0.65	0.69
3	Baseline 2	2.8	3.5	0.70	0.77	0.73
4	Proposed 1	4.3	4.6	0.90	0.80	0.85
5	Proposed 2	4.5	4.8	0.93	0.85	0.89
6						

図 2: 市販 OCR ソフトによる解読結果

	Corpus 1			Corpus 2		
	P	R	F	P	R	F
BASELINE	80.0	70.0	74.7	76.0	78.0	77.0
PROPOSED	85.0	80.0	82.4	83.0	85.0	84.0

図 3: 複数の列にまたがるセルのある表

を作っているので、複数の列がひとつのエクセルの列になっていることがわかる。これは、日本語のフォント・書類にチューニングした結果と思われる。筆者が最初に試したように、日本製のものは、罫線がこまめに入っている表を前提に作られていると推測される。

(a) は、表を解析できなかったのか、セルに文字が格納されておらず、Excel でグラフを作った時のように、セルではない場所に文字が貼付けられている。また、半角と全角が混在している。

(b) はセルに収まっているが、第一列が 1 行下にずれている。

英語で定評のある外国製の OCR ソフトを試したところ、(c) のように正しく変換することができた。

2.3 複数の列にまたがるセルの処理

上記のアルゴリズムでは、複数の列にまたがるセル (LaTeX の\multicolumn に相当するもの。以下では「複数列セル」と略す。) を考慮していない。したがって、図 3 のような表を正しく読むことはできない。

単純な表は正しく読めた外国製の OCR ソフトは右の出力を出した。これも Excel ファイルであるが、文字はセルに格納されておらず、横長の表なのに縦長の出力になっている。「複数列セル」のある表は、市販 OCR ソフトでも読むのが難しいことがわかる。

BASELINE
PROPOSED
Corpus
P R
80.0 70.0
85.0 80.0
1
F
P
74.7
82.4
76.0
83.0
Corpus 2
R
F
78.0 77.0
85.0 84.0

そこで、このような表を読む方法を考えた。多くの場合、複数列セルは、表の一番上の行に出やすく、表の下の方には現れにくいと考えられるので、上 1/3 を隠してヒストグラムを作り、セル分割する。

このセル分割で得られた列境界情報を利用して、複数列セルの行の切れ目を見つける。一部の列境界は文字列にぶつかるので、そこの列境界はないと考え、セルをまとめる。この処理により、図 4 の解読結果が得られた。ここでは、行番号・列番号を行の先頭に出力している。また、列ラベルや行ラベルのセルは出力していない。

3 まとめ

ACL Anthology の論文を対象とした論文 QA を作成しており、その研究の一歩として、これまでの自然言語処理で無視されていた図表を読むためのツールを作成している。本稿では、表を読むのは一見簡単そう

01	01	BASELINE	Corpus_1_P	80.0
01	02	BASELINE	Corpus_1_R	70.0
01	03	BASELINE	Corpus_1_F	74.7
01	04	BASELINE	Corpus_2_P	76.0
01	05	BASELINE	Corpus_2_R	78.0
01	06	BASELINE	Corpus_2_F	77.0
02	01	PROPOSED	Corpus_1_P	85.0
02	02	PROPOSED	Corpus_1_R	80.0
02	03	PROPOSED	Corpus_1_F	82.4
02	04	PROPOSED	Corpus_2_P	83.0
02	05	PROPOSED	Corpus_2_R	85.0
02	06	PROPOSED	Corpus_2_F	84.0

図 4: 提案手法による解読結果

に思えるが、複数の列にまたがるセルが入ると難しいことを指摘し、その解読方法を提案した。

現在、この手法で、ACL Anthology の論文に含まれる表がどれくらい読み取れるかの実験を行っている。グラフを読むツールも現在作成しており、これらについては、別の機会に発表したい。

なお、国立情報学研究所が中心となって進めている「ロボットは東大に入れるか」プロジェクトでは、問題に現れる図表の処理が課題となっている [2] が、本稿で提案した手法は、このプロジェクトにも転用できるであろう。

謝辞

本研究は JSPS 科研費 26330366 の助成を受けたものです。

参考文献

[1] 東中竜一郎, 貞光九月, 内田渉, 吉村健. しゃべってコンシェルにおける質問応答技術. NTT 技術ジャーナル, Vol. 25, No. 2, 2013.

[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. 言語処理学会年次大会, 2015.

[3] 市野順子, 箕牧数成, 山口和泰, 垣智, 東郁雄, 古田重信. 図表検索のための図表情報自動抽出の試み. 情報処理学会　デジタルドキュメント 32-19, 2002.

[4] 磯崎秀樹. PDF 中の TEX 記号の復元と ACL Anthology への適用. 言語処理学会年次大会, 2013.

[5] 磯崎秀樹, 東中竜一郎, 永田昌明, 加藤恒昭. 質問応答システム. コロナ社, 2009.

[6] Hideki Isozaki, Katsuhito Sudoh, and Hajime Tsukada. NTT's Japanese-English cross-language question answering system. In *Working Notes of the NTCIR Workshop Meeting (NTCIR)*, pp. 186–193, 2005.

[7] Leslie Lamport. *A Document Preparation System LATEX second edition*. Addison-Wesley Publishing Company, 1994.

[8] 中野康明, 藤澤浩道, 国崎修, 岡田邦弘, 花野井歳弘. 文字認識と協調した表形式文書の理解. 電子情報通信学会論文誌, Vol. J69-D, No. 3, 1986.

[9] スティーヴン・ベイカー. IBM 奇跡のワトソンプロジェクト：人工知能はクイズ王の夢を見る. 早川書房, 2011.

[10] 田端康人, 鶴岡信治, 木村文隆, 三宅康二. 表の構造理解のための罫線抽出と領域分け. 電子情報通信学会技術研究報告 PRU90, pp. 68–80, 1990.