とにかく booktabs を使おう

東京大学　情報理工学系研究科　講師　松井勇佑

初版 2022 年 9 月 5 日 / 最終更新 2023 年 1 月 23 日

- 本資料の GitHub リポジトリ： https://github.com/matsui528/use_booktabs_anyway
- 著者のウェブページ：http://yusukematsui.me

1　はじめに

論文に表を記載する際、booktabs パッケージを使いましょう。booktabs は通常の TeX の表に対し適切な線の使用やスペースの配置を補助してくれるパッケージです。booktabs を使うだけで表がシュッと綺麗になります。それだけでなく、booktabs 流の表の作り方（行を基本とする思考、縦線を入れない、など）を考えていくと、論理的にわかりやすい表になります。以下の 2 つが基本的な文献です。

- 公式ドキュメント。細かいオプションなどは CTAN の公式ドキュメントを参照しましょう。
- Markus Püschel, “Small Guide to Making Nice Tables”. 本資料と同じ立ち位置の、booktabs 紹介記事です。

本文章は上記文献と松井の経験則をベースにしています。間違いや、より良い方法もあると思います。提案やコメントやミスの指摘などがあればぜひ issue でお願いします。

2　基本的な使い方

次の 2 つの表を考えましょう。

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5

表 1　通常の表

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5

表 2　booktabs を用いた表

左は通常の TeX の機能でなんとなく作った表です。多分最初に作る表はこんな感じになると思います。右は同じ内容を booktabs を使って整えた例です。シュッとしていてカッコいいですよね？ ここで、左側の表の tex のソースは以下です。

```
% 「表 1 通常の表」
\begin{tabular}{|c|c|c|} \hline
    名前 & 身長 (cm) & 体重 (kg) \\ \hline
    オデロ・ヘンリーク  & 173 & 62.2 \\ \hline
    オリファー・イノエ  & 178 & 69.5 \\ \hline
\end{tabular}
```

一方で、右側の booktabs を用いたソースは次のようになっています。

```
% 「表 2 booktabs を用いた表」
\begin{tabular}{@{}lll@{}} \toprule
    名前 & 身長 (cm) & 体重 (kg) \\ \midrule
    オデロ・ヘンリーク  & 173 & 62.2 \\
    オリファー・イノエ  & 178 & 69.5 \\ \bottomrule
\end{tabular}
```

ここで左の表を右の表に変換する機械的なルールを紹介します。以下の手続きを行えば OK です。

1. booktabs パッケージをインポートする: ソースコード中の最初のほうで`\usepackage{booktabs}`としてパッケージをインポートしましょう。これを忘れると当然うまくいきません。
2. 縦線を取り除く：表 1 では列指定オプションは `{|c|c|c|}` となっています。ここから縦線を取り除きます。すなわち、`{ccc}` とします。基本的に表の縦線は必要ない場合が多いです。縦線がどうしても必要な場合は、表を分割するほうが良い場合が多いです。
3. 左揃えにする：すなわち、`{ccc}` だったものを `{lll}` とします。まずは全てを左揃えにすることをオススメします。そのうえで見栄えが悪ければ中央や右揃えを考えましょう。
4. 魔法のスペーサー記号をつける： `@{}` という魔法のスペーサー記号を列指定オプションの両端につけます。すなわち、`{lll}` だったものを `{@{}lll@{}}` とします。これを忘れる人が多いです。このスペーサーをつけると、表の両端の余分な空白がなくなり、見栄えが良くなります。スペーサーがない不揃いな例が表 3 で、スペーサーがある綺麗な例が表 4 です。
5. 横線を減らし rule 記号に変える: 横線`\hline`は全て消し、「一番上」に`\toprule`、「ヘッダ直後」に`\midrule`、「一番下」に`\bottomrule` を書きましょう。

これでおしまいです。これをやる・やらないだけで全く見栄えが違います。本資料で言いたいことの 8 割はこれで終わりです。以下は、発展的な内容の紹介です。

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5

表 3　`@{}` を使っていない、不揃いなスペーシングの表

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5

表 4　`@{}` を使っている、綺麗なスペーシングの表

3　発展編

発展編として、良い表を作るための指針や様々な技術について紹介します。

3.1　行志向

行が一つの単位であり、表とは**行の集合である**ということを年頭に置くとよいです。ある一度の測定や、ある一つのデータが、一つの行です。各行は独立しています。そして、各行は、概念として「同じレベルのもの」です。プログラム的に言えば「同じ型のインスタンス」です。表を「ヘッダ（型の定義）」と「行の集合（インスタンスの集合）」だけで構成できるとわかりやすいです。

表 2 の例では、第一行のヘッダ部分に「名前」など列の説明が記載され、第二行以降の本体部分は「一人のデータが一行」になっています。ここで、各行は、順番を並び替えても問題ありません（オデロさんとオリファーさんを入れ替えても大丈夫）。実際に表を作るときは並びに気を使い身長順にソートしたりしますが、セマンティックには、各行の入れ替えが可能です。そのような表を目指すとよいです。表本体部分について、並び変えるとおかしくなる場合は、表の構成を考え直したり、グラフで表現できないか考えてみるといいです。

booktabs の三種類の横線（`\toprule`, `\midrule`, `\bottomrule`）はこの行志向の表の構成に対応します。一番上が`\toprule`で、一番下が`\bottomrule`で、あとは「ヘッダ」と「行の集合」の間に`\midrule`をおきます。そのスタイルが基本です。

上記はもちろん原則です。考えられる例外としては、「合計」とか「平均」の行を最初や最後に置く場合が挙げられると思います（表 5）。合計や平均は他の行に依存するため、他の行から独立していません。また、合計や平均は表の最初か最後の行に書くことが多いでしょう。このような例では、表 6 のように、わかりやすくするために「平均」の上に横線を引くこともあると思います。しかし横線の引きすぎには注意してください。

表を作るということは、行志向の原則を可能な限り守りつつ、「読者への情報提示をわかりやすくするにはどうすればよいか」「限られた紙面スペースをどう有効活用するか」ということを模索して作っていくものだと言えます。

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5
オイ・ニュング	167	52.1
平均	172.7	61.3

表 5　最後に平均が記載される例

名前	身長 (cm)	体重 (kg)
オデロ・ヘンリーク	173	62.2
オリファー・イノエ	178	69.5
オイ・ニュング	167	52.1
平均	172.7	61.3

表 6　最後に平均が記載し、またその直前に横線を加える例

3.2　行のグループ化

表を見やすくする一つの方法が行のグループ化です。表 7 にもともとの表を示します。ここでは Hoge 手法と提案手法について、精度と速度を報告しています。また、どちらの手法もパラメータとして k があり、k を変化させて値を報告しています。この表は「一行が一つの観測」となっており、行志向の適切な表です。

一方で、一番左の列には Hoge や Ours が並んでおり、読者に提示する情報としては冗長です。この場合、表 8 に示すように、冗長な項目を削り行をグループ化してしまうとよいです。こうすることで、表としての情報は維持したまま、文章量を削ることが出来ました。

表 8 で十分であり、これで終わりで OK です。一方で、もし望むのであれば Hoge や Ours を`\multirow`を用いて中心に寄せてもよいです。その例が表 9 です。この場合、横線がないとわかりづらいので、`\midrule`で横線を追加しています。表 8 と表 9 のどちらが良いかは状況に依存すると思います。もし横線をたくさん引かないとわかりづらい表になっているとすれば、それは表を分割するほうが分かりやすい可能性があります。

手法	k	速度 (ms)	精度 (Recall)
Hoge	16	0.32	0.21
Hoge	32	0.61	0.44
Ours	9	0.47	0.26
Ours	18	0.99	0.77

表 7　もともとの表

手法	k	速度 (ms)	精度 (Recall)
Hoge	16	0.32	0.21
	32	0.61	0.44
Ours	9	0.47	0.26
	18	0.99	0.77

表 8　行をグループ化した場合。これで OK。

手法	k	速度 (ms)	精度 (Recall)
Hoge	16	0.32	0.21
	32	0.61	0.44
Ours	9	0.47	0.26
	18	0.99	0.77

表 9　さらに中心に寄せ横線を引いた場合。これでも OK。横線の引きすぎに注意。

また、行のグループ化は複数回行うこともできます。例を表 10 に示します。ここでは大学近辺のラーメン屋さん情報をまとめています。イメージとしては一列目でソートし冗長な項目を消す、二列目でソートし冗長な項目を消す、と繰り返していくものです。やりすぎると見にくくなるので注意しましょう。

最寄り駅	店舗名称	メニュー品目	値段（円）
本郷三丁目	海手	なつラーメン	700
		まぜ麺 X	850
	IBASA	ラーメン	700
		冷たいラーメン	650
東大前	用心麺	ラーメン	700
		つけ麺	800

表 10　行のグループ化を二度行った例

3.3　行の階層化

次に、行を階層化するテクニックを紹介します。表 11 では、お店で販売している品目の情報がまとめられています。これも各行が独立しており、良い表です。ここで、種別の情報は冗長なので、「行のグループ化」によりまとめたものが表 12 です。これで OK です。

ここでさらに、種別情報を用いて品目を階層化した例が表 13 です。ここでは種別の列を削除し、それを各品目の上位のものとして表示しています。このように、列を削除して要素を字下げすることで、行の階層化を達成できます。ここで字下げは単純に空白を入れることで実現しています。詳しくは TeX のソースを見てみてください。

このような階層化が役立つ例を一つ紹介します。神奈川県の市の経済規模を調査したいとしましょう。表 14 に、神奈川県総計の情報、および 3 つの政令指定都市の情報が示されています。比較のため、石川県と富山県の情報も載っています。この表も、各行が各地域を表し独立しているため、良い表です。一方で、「総計」は県全体を表すのに対し、「横浜市」は市単体を表すため、表現の粒度が行によって異なり、直感的に理解しづらいかもしれません。

ここで、行の階層化を行った例が表 15 です。これによりずっと理解が簡単になりました。ここではたとえば、横浜市は「市」であるにもかかわらず、面積では 10 倍の石川「県」よりも人口が 3 倍多いことが見てとれます。

注意として、行のグループ化も階層化も、見やすさのために行っているだけだということを意識してください。すなわち、いつでも「各行が一つのデータ」である「もともとの表」に戻れるような表にするとよいです。

品目	種別	値段（円）	売り場（階）
豚肉	肉	300	2
牛肉	肉	500	2
トマト	野菜	100	3
キュウリ	野菜	200	3
キャベツ	野菜	30	4

表 11　もともとの表

品目	種別	値段（円）	売り場（階）
豚肉	肉	300	2
牛肉		500	2
トマト	野菜	100	3
キュウリ		200	3
キャベツ		30	4

表 12　種別についてグループ化した例

品目	値段（円）	売り場（階）
肉		
豚肉	300	2
牛肉	500	2
野菜		
トマト	100	3
キュウリ	200	3
キャベツ	30	4

表 13　さらに階層化した例

県	市	面積（km^2）	人口（人）
神奈川県	総計	2,416	9,237,000
神奈川県	横浜市	437	3,774,000
神奈川県	川崎市	143	1,542,000
神奈川県	相模原市	328	726,000
石川県	総計	4,186	1,119,000
富山県	総計	4,247	1,018,000

表 14　もともとの表

地域	面積（km^2）	人口（人）
神奈川県		
総計	2,416	9,237,000
横浜市	437	3,774,000
川崎市	143	1,542,000
相模原市	328	726,000
石川県	4,186	1,119,000
富山県	4,247	1,018,000

表 15　行を階層化した例

3.4　列の階層化

次に、「列の階層化」を紹介します。列は全て違うものを表すので、列をグループ化することはありませんが、階層化することは出来ます。その例を表 16 に示します。ここでは手法の最小エラー、平均エラー、最大エラーを表示しています。これでＯＫなのですが、いちいち毎回ヘッダに「エラー」と書くのは冗長だとも言えます。そのような場合、ヘッダの項目を階層化することが出来ます。

階層化した例を表 17 に示します。ここでは新たな横線を導入することで、もともと別々だった列をまとめることに成功しました。ここで、「エラー」と書かれる部分について、複数にまたがる要素は`\multicolumn`コマンドで作成可能です。また、「部分横線」は`\cmidrule`というコマンドで記述することが出来ます。

手法	最小エラー	平均エラー	最大エラー
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 16　もともとの表

	エラー		
手法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 17　列を階層化した例

「列の階層化」も、何度も行うことが出来ます。その例を表 18 に示します。ここでは様々な県の情報を表示しています。気温や降水量について、階層化を行うことで見やすくなっています。また、単位の記述を上部に集約するなどして、横方向にだいぶ縮めることが出来ています。階層化させていない項目（県名や人口密度）も同じ表に表示させることが可能です。

	気温（°C）			降水量（mm）		交通の便		
県名	最高	平均	最低	8 月	12 月	新幹線	空港	人口密度（人/km^2）
石川県	32	20	-1	179.8	304.7	✓	✓	267
静岡県	27	23	5	250.9	63.0	✓	✓	461
沖縄県	33	28	15	175.4	104.4		✓	643

表 18　列の階層化を複数回行った表

3.5　部分横線：cmidrule

ここで、「列の階層化」で登場した\cmidruleについて知っておきましょう。表 17 のソースコードは以下のようになります。

```
\begin{tabular}{@{}llll@{}} \toprule
           & \multicolumn{3}{c}{エラー} \\ \cmidrule(l){2-4}
    手法    & 最小 & 平均 & 最大            \\ \midrule
    Isomap & 0.23 & 0.44 & 0.92          \\
    LLE    & 0.10 & 0.73 & 1.82          \\ \bottomrule
\end{tabular}
```

\cmidrule(l){2-4}において、「l」の部分は、横線の両端を削るかどうかを意味しています。l であれば左を、r であれば右を、lr であれば両端を削ります。この少しの削りで、表の見栄えを調整します。その例を表 19 から表 22 に記載しておきました。微妙な違いなのですが、知っておくと便利です。{2-4} の部分は、対応する列数です。これはいじりながら調整すればよいでしょう。

	エラー		
手法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 19　cmidrule(){2-4}

	エラー		
手法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 20　cmidrule(l){2-4}

	エラー		
手法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 21　cmidrule(r){2-4}

	エラー		
手法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 22　cmidrule(lr){2-4}

ちなみに、表 18 では\cmidurleは「\cmidrule(lr){2-4} \cmidrule(lr){5-6} \cmidrule(lr){7-8}」のように全ての線に「lr」を付けて、線の両端を削る形で記述されていました。これを削らない場合を表 23 に示します。ここでは\cmidrule(){2-4}のようにすべての「lr」を取っています。よって、「気温」「降水量」「交通の便」の間の線がつながってしまっていることがわかります。ここでは適切に空白を入れるほうがわかりやすい表になります。

	気温（°C）			降水量（mm）		交通の便		
県名	最高	平均	最低	8 月	12 月	新幹線	空港	人口密度（人/km^2）
石川県	32	20	-1	179.8	304.7	✓	✓	267
静岡県	27	23	5	250.9	63.0	✓	✓	461
沖縄県	33	28	15	175.4	104.4		✓	643

表 23　表 18 で cmidrule の削りを適切に設定しなかった場合

3.6　列を行に移す

列の情報を行に移すことで表がスッキリする場合があります。この技術はオススメ**では無い**のですが、現実的に遭遇する場面が多いため紹介します。例を用いて説明します。表 24 がもともとの表です。ここでは 2 つの手法を 3 つのデータセットで評価し、実行時間を報告しています。各行が 1 つの観測に対応する、良い表です。手法に関して行をグループ化しています。この表はこれでオシマイでいいのですが、以下の二点の要請があり得ます。

- 同じ Dataset で手法を見比べたいが、値の位置が離れている。例えば MNIST 上での k-means (10.2) と Ours (8.3) が離れている。
- Method や Dataset が増えていくと、表が縦長になってしまう。横長にしたい。

このような場合、どうするとよいでしょうか？

一つの列（ここでは Dataset）を行方向に移すということが考えられます。「一つのデータ（一つの行）」を、「手法・データセット・実行時間」ではなく、「手法・データセット 1 の実行時間・データセット 2 の実行時間・データセット 3 の実行時間」と考え直すということです。そうしたものを表 25 に示します。これにより、2 つの要請である「同じデータセットでの値の位置を近くする」「横長にする」はクリアすることが出来ました。一方で、ヘッダ部分が明らかに冗長になってしまっています。

これに対し「列の階層化」を適用したものが表 26 です。だいぶ良くなりました。ここでオシマイでも OK です。しかしここでは「Dataset=」という記述がヘッダに 3 度登場し、冗長です。これをさらにスッキリさせる 2 つのアイデアが表 27 と表 28 です。

Method	Dataset	Runtime (ms)
k-means	MNIST	10.2
	ImageNet	45.3
	Places	57.1
Ours	MNIST	8.3
	ImageNet	39.1
	Places	82.3

表 24　もともとの表

Method	Runtime (ms) for Dataset=MNIST	Runtime (ms) for Dataset=ImageNet	Runtime (ms) for Dataset=Places
k-means	10.2	45.3	57.1
Ours	8.3	39.1	82.3

表 25　Dataset の列を行に移したもの。ヘッダが冗長。

	Runtime (ms)		
Method	Dataset=MNIST	Dataset=ImageNet	Dataset=Places
k-means	10.2	45.3	57.1
Ours	8.3	39.1	82.3

表 26　列を階層化したもの。マシになったが、「Dataset=」の部分がまだ冗長。

表 27 では、表 26 から「Dataset=」の部分を取り除いています。これで表は見やすくなりました。今回の表の場合はこれでも意味が通じるでしょう。一方で、ここでは「Dataset=」という記述がどこにも書かれていないことになるため、「MNIST」の部分が一体何なのかわかりません。よって、このパターンは Runtime 以下の階層が説明なしに伝わる場合にしか使えないでしょう。

表 28 では、表 26 から「Runtmie (ms)」を取り除き、その位置に「Dataset」を記入しています。これが一番スッキリします。一方で、この場合は表の値が何なのかという情報が抜け落ちるため、キャプションに「これは Runtime (ms) の表です」という記述を追加する必要があります。これは、「表が単一種類の値のみで構成されている場合、その値の説明をキャプションで行うことで、ヘッダに表示する情報を減らし、表をスッキリさせることができる場合がある」ということです。ちなみに、この単一種類の値の表は、本資料の他の表と違う特長を持ちます。すわなち、ヘッダが要素の説明になっていません。個人的にはこの方式はあまり好きではなく、ヘッダが要素説明になっているほうがいいと思います。一方で、スペースの要請からどうしてもそのようにする場面もあるでしょう。

	Runtime (ms)		
Method	MNIST	ImageNet	Places
k-means	10.2	45.3	57.1
Ours	8.3	39.1	82.3

表 27　「Dataset=」を省略した場合。見やすくなったが、「Dataset=」の情報はどこにも記述できない。

	Dataset		
Method	MNIST	ImageNet	Places
k-means	10.2	45.3	57.1
Ours	8.3	39.1	82.3

表 28　「Runtime (ms)」を省略した場合。一番スッキリするが、「Runtime (ms) の表である」という情報をキャプションに記述する必要がある。

さて、そもそも今回色々な表を考えたのは、Method 方向・Dataset 方向のどちらにも注目したかったからです。表は行と列しかないので、二軸に注目する情報を提示する場合は、表 27 や表 28 のようにどうしてもマトリクス的な表現になってしまいます。よって、「一つの行は一つの観測」の原則からだんだん離れていってしまいます。そのような場合、グラフの使用も検討してください。同じ情報をグラフで提示したものが図 1 です。ここでは「あるデータセットにおける手法の比較」も「ある手法に関してデータセットを変えた際の比較」も可能です。また今回は、データセット・手法を増やしてもグラフ上のスペースは変わりません。さらに、エラーバー追加や箱髭図への変更も可能です。一方で、グラフを用いる弱点は具体的な数値が表現されないことです。例えば既存手法との数値の比較が必要な際はグラフは使いにくいかもしれません。表とグラフのメリットデメリットを考えながら、最も有効な情報表現を考えていくとよいでしょう。

3.7　表の作成で悩んだら

表の作成で悩んだ場合、以下を順番にやってみましょう。

1. 「一行が 1 つのデータ」の原則になるまで、表を分解する。とても縦に長くなっても OK。
2. 一番重要な項目について、「行のグループ化」を繰り返す。必要に応じて「行の階層化」も行う。
3. 必要に応じて「列のグループ化」を行う。
4. それでもまだおさまりが悪ければ、「列を行に移す」を行い、「行の階層化」を行う。冗長な記述を省略する。
5. それでもまだおさまりが悪ければ、グラフの使用を検討する。

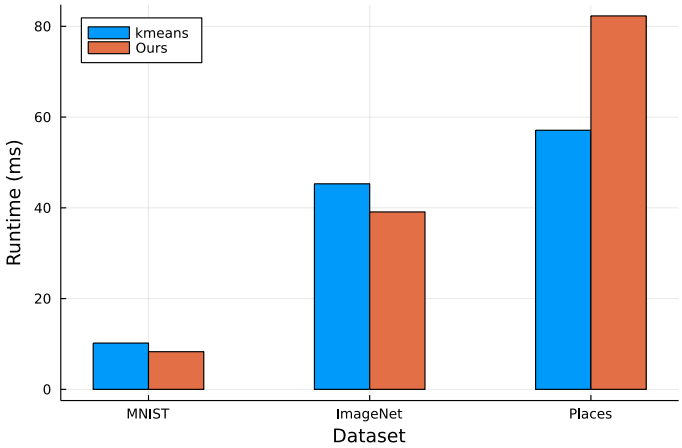

図 1　表 27 や表 28 をグラフにしたもの。

4　ケーススタディ

さて、いくつか例をあげて、表をどのように綺麗にするか見ていきましょう。まずは表 29、表 30、表 31 を考えます。ここではパラメータを変えた際の各手法の精度を報告しています。これらは全て同じ情報を示そうとしていますが、ありがちなミスを含んでいます。

まず表 29 から見てみましょう。ここでは、パラメータ T を変化させたときの、ResNet50 と提案手法のスコアの変動を表現しています。この表は一見こぎれいにまとまっているように見えます。しかしよく見ると、ヘッダであるべき一行目はヘッダになっていないことがわかります。たとえば ResNet50 の上には「Method」という情報が来るべきですが、ここでは T になってしまっています。また、「0.32」の上は「1」になっており、これも説明になっていません。なので、これは「ヘッダ + 行の集合」のイケている表にはなっていません。加えて、この表には要素の情報（精度であるということ）が記載されていません。すなわち、この数字が何なのかわかりません。キャプションに書く必要があります。このような小さい表ではキャプションを使わずに表中に全て記載するほうがスッキリする可能性が高いです。

同じ情報を、表 30 のように書いてしまうことも多いでしょう。ここでは、左上のマスに「斜め線」と「両軸の情報」を記載しています。この方式は小学校や中学校で習うため、ついこう書いてしまう人が多いようです。しかし、これもまた、「ヘッダ + 行の集合」形式になっていません。加えて、この方式を書くときは縦線を記入してしまうことが多いようです。

あるいは、表 31 のように書くかもしれません。これは表 29 に比べて少し良くなり、0.32 の上は「$T = 1$」であることがわかります。しかし、「数字が何なのか」は、やはりわかりません。

T	1	4	16
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 29　ありがちな例その 1

Method \ T	1	4	16
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 30　ありがちな例その 2

	$T = 1$	$T = 4$	$T = 16$
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 31　ありがちな例その 3

さて、この表を良くするにはどうすればいいでしょうか？ まず、「一行が一つのデータ」になるまで分解してみます。ここでの「一つのデータ」とは、「ある手法で、あるパラメータ T で、測定結果である精度（mAP）」になります。その結果が表 32 です。ここでは、各要素が「精度」であると言う点をちゃんとヘッダに記述できています。これで終わりでもいいのですが、もう少し考えてみましょう。実はこの表は表 24 と同じ構造をしています。なので、「T の列を行に移す」「列を階層化する」を行い、表 33 のようにまとめることが出来ます。これは表 24 を表 26 にしたものに相当します。ここでは $T = 1$ といった記述は十分短いなので、この表が一番見やすく情報を提示できているでしょう。表 31 に比べると、「mAP」という情報を表の中に記述できており、また「Method」という説明も追加されています。

ここで、表 27 や表 28 のようにさらに省略を行うとどうなるでしょうか？ それを表 34、表 35 に示します。表 34 では、「$T =$」の部分を省略しています。今回はこの省略は有効ではありません。「1, 4, 16 が何なのか」がわからなくなってしまっています。なのでこれはやめておいたほうがいいですね。表 35 では、「mAP」の部分を省略しています。この場合、その情報をキャプションに書く必要があります。今回の例ではここまで省略する必要はなさそうです。

ちなみに、今回もグラフにすることが出来ます。それを図 2 に示します。ここでは T は変化させることができる連続量なので、棒グラフではなく折れ線グラフにしています。グラフと表のどちらがいいかは、状況に応じて決めてください。

さて、別の例も見てみましょう。次は表 36 を見てます。この例は表 29 に似ています。おさまりは良く見えますが、一行目がヘッダになっていない、イマイチな表です。しかも、表 29 では一列目を「Method」としてまとめられたのに対し、この例ではそれに相当するう

Method	T	mAP
ResNet50	1	0.32
	4	0.54
	16	0.77
Ours	1	0.41
	4	0.81
	16	1.23

表 32　「一行が一つのデータ」になるまで分解した場合

	mAP		
Method	$T = 1$	$T = 4$	$T = 16$
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 33　T 列を行に移し、列を階層化した場合。今回はこれが一番見やすい。

	mAP		
Method	1	4	16
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 34　「$T =$」を省略した場合。ここでは 1, 4, 16 の意味が通じなくなってしまっており、よくない表になっている。

	T		
Method	1	4	16
ResNet50	0.32	0.54	0.77
Ours	0.41	0.81	1.23

表 35　「mAP」を省略した場合。「mAP の表である」という記述をキャプションに書く必要がある。

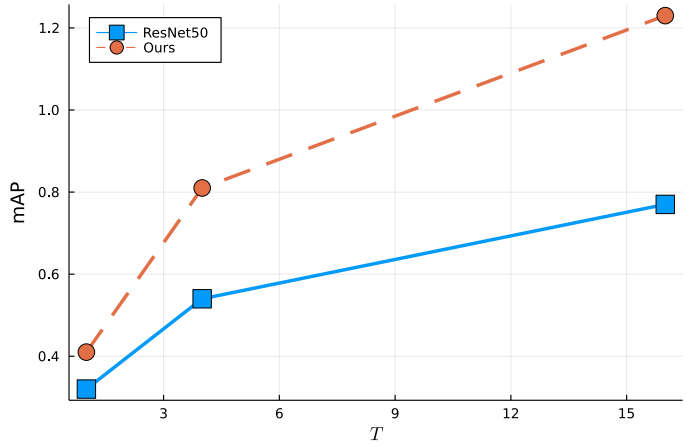

図 2　表 33 をグラフにしたもの。

まい単語がありません。どうすればよいでしょうか。今回の場合は実は、じっと眺めていると、行と列を全て入れ替えた表 37 のほうが、「各行が 1 つのデータ」に対応するスッキリした表になっていることがわかります。なので、行志向を目指す考えのもとでは表 36 のほうが良い表であると言えます。一方で、ここではヘッダ部分の項目が長すぎて、表中に空白が目立っています。スペースを有効活用するためにどうするかは、腕の見せどころです。例えば単位を改行したりするといいかもしれません。

T	1	4	16
Runtime (sec)	102	110	159
Memory (B)	100	200	300

表 36　もともとの表

T	Runtime (sec)	Memory (B)
1	102	100
4	110	200
16	159	300

表 37　行と列を入れ替えた表