总而言之来使用 booktabs 吧

东京大学　情报理工学系研究科　讲师　松井勇佑

初版 2022 年 9 月 5 日 / 更新日期 2023 年 1 月 23 日

- 本资料的 GitHub 仓库：https://github.com/matsui528/use_booktabs_anyway
- 作者的主页：http://yusukematsui.me

1　序言

在论文中插入表格时，来使用 booktabs 的宏包吧。booktabs 是在一般 Tex 表格基础上，可对线和宽度进行适当调整的宏包。仅通过使用 booktabs 就能轻松让表格变得整洁美观。不仅如此，按照 booktabs 的思路（以行为本、不插入竖线等）来制作表格的话，会使制成的表格更加清晰易懂。以下是两篇基础文献。

- 官方文档。细节的设定等请参照 CTAN 的官方文档。
- Markus Püschel, "Small Guide to Making Nice Tables". 与本资料同样是介绍 booktabs 的文章。

本文以上述文献和我的个人经验为基础，可能存在错误或更好的方法。如果有感想建议或者有想指出的错误的话，请务必在 issue 提出。

2　基本使用方法

请看下面两个表格。

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5

表 1: 普通表格

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5

表 2: 使用 booktabs 的表格

左边是使用 Tex 的一般功能制作出的表格，估计大家最开始做的表格都是这种感觉的吧。右边是使用 booktabs 对相同内容进行整理后的例子。是不是变得整洁又大方了呢？左侧表格的 tex 代码如下所示。

```
% "表 1 普通表格"
\begin{tabular}{|c|c|c|} \hline
   姓名 & 身高 (cm) & 体重 (kg) \\ \hline
   奥迪罗 · 汉宁古  & 173 & 62.2 \\ \hline
   奥利法 · 伊诺埃  & 178 & 69.5 \\ \hline
\end{tabular}
```

与之相对地，右侧使用 booktabs 的表格的代码如下所示。

```
% "表 2 使用 booktabs 的表格"
\begin{tabular}{@{}lll@{}} \toprule
   姓名 & 身高 (cm) & 体重 (kg) \\ \midrule
   奥迪罗 · 汉宁古  & 173 & 62.2 \\
   奥利法 · 伊诺埃  & 178 & 69.5 \\ \bottomrule
\end{tabular}
```

接下来将介绍将左边表格变换成右边表格的机械化规则。按照以下步骤来做就可以实现了。

1. 导入 booktabs 的宏包：在代码起始位置添加`\usepackage{booktabs}`来导入宏包。忘记这一步的话肯定是无法顺利进行下去的。
2. 去除竖线：表**??**里的列的设定为 `{|c|c|c|}`。要通过修改这个地方来把竖线去掉，也就是说，将它改为 `{ccc}`。表格里基本上是不需要竖线的。无论如何都需要竖线时，大多数情况下把表格进行分割会更好。
3. 设置为左对齐：也就是把 `{ccc}` 改为 `{lll}`。推荐首先全部进行左对齐，如果觉得不够美观的话再考虑居中或者右对齐。
4. 添加神奇的间隔符：把神奇的间隔符 `@{}` 添加到列的设置的两端。即把 `{lll}` 改成 `{@{}lll@{}}`。很多人容易忘记这一步。添加了这个间隔符的话，可以去掉表格两端多余的空白，使表格看上去更加美观。未添加间隔符的不够整洁的表格如表**??**所示，添加了间隔符后变美观的表格如表**??**所示。
5. 删除横线并改成 rule 符号：把横线`\hline`全部删除，“上框线”用`\toprule`、“表头后”用`\midrule`、“下框线”用`\bottomrule`。

这样就完成了。只是通过增加或删除操作就能带来截然不同的视觉效果。到这里本资料所要传达的内容已经实现了八成。接下来是扩展内容的介绍。

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5

表 3: 未使用 `@{}` 的不够整洁的表格

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5

表 4: 使用 `@{}` 后变美观的表格

3 扩展篇

扩展篇将介绍制作优秀表格的方针和各种技术。

3.1 以行为本

建议时刻谨记**以行为一个单位**，表格是**行的集合**这件事。某次测量或某条数据就是一行，各行之间是独立的。其次，每一行在概念上都是“同一级别的东西”。用编程语言来形容的话就是“同一类型的实例”。如果一个表格能够只由“表头（类型定义）”和“行的集合（实例的集合）”构成，那它就是一个易于理解的表格。

在表**??**的例子中，第一行的表头部分是“姓名”等对列的说明，第二行以后的主体部分满足“一个人的数据为一行”原则。在这个例子中，即使改变每行的排列顺序也没有问题（交换奥迪罗和奥利法的顺序也没关系）。虽然在实际制作表格时，以身高为基准进行了排序，但从含义上来讲，交换各行的顺序也没有关系。我们的目标就是能够制作出这样的表格。如果改变表格主体部分的排列顺序会感到不自然的话，重新考虑表格的构成或者考虑能否用图表来表示会比较好。

booktabs 的三种横线(`\toprule`, `\midrule`, `\bottomrule`)与这种以行为单位的表格的构成是相匹配的。最上面一条线是`\toprule`，最下面一条线是`\bottomrule`，然后就是在“表头”和“行的集合”之间插入`\midrule`。这种是基本的格式。

当然上述只是原则上的格式，也存在例外情况。比如将“合计”或“平均”那行放到最初或最后（表**??**）。由于合计或平均与其他行相关，所以它们不是独立于其他行的。很多情况下，会将合计或平均放到表格的最初或最后。在这个例子中为了便于理解，也可以像表**??**那样在“平均”之上画一条横线。只不过需要注意不要画太多的横线。

在制作表格时，除了遵守“以行为本”的原则外，还要在制作过程中逐渐摸索怎样才能更加便于读者理解，以及怎样才能充分利用有限的纸面空间。

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5
奥伊 · 宁格	167	52.1
平均	172.7	61.3

表 5: 把平均放在最后的例子

姓名	身高 (cm)	体重 (kg)
奥迪罗 · 汉宁古	173	62.2
奥利法 · 伊诺埃	178	69.5
奥伊 · 宁格	167	52.1
平均	172.7	61.3

表 6: 把平均放在最后并在其上面加了横线的例子

3.2 行的分组

让表格变得直观的方法之一就是行的分组。表??是原本的表格，分别展示了 Hoge 的方法和我们的方法下的速度和精度。此外，每个方法还有一个参数 k，表中也展示了 k 的值的变化。这个表格是符合“一行是一个对象”原则的以行为本的表格。

另一方面，最左列的 Hoge 和 Ours 并列在一起，对于读者来说存在信息冗余。这种情况下，像表??这样，将冗余的项目删除，对行进行分组会更好。这种方式可以在保证表格信息量的前提下达到缩减字数的效果。

实现表??这种形式就已经可以了。此外，还可以在 Hoge 和 Ours 中间用`\multirow`将它们区分开，如表??所示。在这个例子里，由于没有横线的话会比较难懂，所以用`\midrule`添加了一条横线。表??和表??哪个更好，主要取决于实际情况。如果一个表格不加很多横线就会变得很难懂的话，把它分割成几个表格可能会更加便于理解。

方法	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: 原本的表格

方法	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: 将行进行分组后的表格。这样就可以。

方法	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: 为了更加集中额外加了一条横线。这样也可以。要注意不要加太多横线。

此外，可以进行多次行的分组和合并，如表??所示。这是大学附近拉面店的整合信息。整体思路是先对第一列的项目进行排序然后去除冗余，再对第二列项目进行排序然后去除冗余，如此反复。请注意如果过度合并的话可能会变得难以阅读。

最近的车站	店铺名称	菜品	价格（元）
本乡三丁目	海手	natsu 拉面	700
		拌面 X	850
	IBASA	拉面	700
		冷面	650
東大前	用心面	拉面	700
		蘸面	800

表 10: 进行了两次行的分组合并的例子

3.3 行的分级

接下来介绍行的分级技巧。表??中总结了店里售卖的商品信息。每行之间相互独立，是一个优秀的表格。由于这里存在类别信息的冗余，所以使用“行的分组”会变得更加整洁，像表??这样就可以。

在此基础上，利用类别信息对商品进行分级的例子如表??所示。在这里删掉了类别那一列，把它放到了各个商品的上一级。像这样通过删除列来减少字数，以达成行的分级。这里只通过插入空白就实现了字符的缩进，具体请看 Tex 的代码。

商品	类别	价格（元）	位置（层）
猪肉	肉	300	2
牛肉	肉	500	2
番茄	蔬菜	100	3
黄瓜	蔬菜	200	3
卷心菜	蔬菜	30	4

表 11: 原本的表格

商品	类别	价格（元）	位置（层）
猪肉	肉	300	2
牛肉		500	2
番茄	蔬菜	100	3
黄瓜		200	3
卷心菜		30	4

表 12: 根据类别分组后的例子

商品	价格（元）	位置（层）
肉		
猪肉	300	2
牛肉	500	2
蔬菜		
番茄	100	3
黄瓜	200	3
卷心菜	30	4

表 13: 进一步分级后的例子

接下来介绍一个分级可以带来显著效果的例子。假设我们想要调查神奈川县各个市的经济规模。表**??**展示了神奈川县的统计信息和其中三个行政市的信息。为了比较，还列出了石川县和富山县的信息。这个表格每行都能独立表示各个地域的信息，也是一个优秀的表格。与此同时，表格中既有表示县整体情况的“总计”，也有“横滨市”这样单独表示一个市的情况。由于各行所表示的级别不同，所以直观上可能难以理解。

因此将行进行分级后的例子如表**??**所示。这种方式使理解变得容易了许多。比方说，很容易能看出，即使横滨市是“市”，它的人口也是面积是它 10 倍的石川“县”的 3 倍多。

需要注意的是，进行行的分组和分级都只是为了便于观看。也就是说，制作出的表格要能随时都能恢复成“每行是一条数据”的“原本的表格”。

县	市	面积（km^2）	人口（人）
神奈川县	总计	2,416	9,237,000
神奈川县	横滨市	437	3,774,000
神奈川县	川崎市	143	1,542,000
神奈川县	相模原市	328	726,000
石川县	总计	4,186	1,119,000
富山县	总计	4,247	1,018,000

表 14: 原本的表格

地区	面积（km^2）	人口（人）
神奈川县		
总计	2,416	9,237,000
横滨市	437	3,774,000
川崎市	143	1,542,000
相模原市	328	726,000
石川县	4,186	1,119,000
富山县	4,247	1,018,000

表 15: 将行进行分级后的例子

3.4 列的分级

接下来介绍“列的分级”。由于每列表示的都是完全不同的东西，所以无法进行列的分组。不过可以进行列的分级，其中一个例子如表**??**所示。在这个表中展示了不同方法的最小误差、平均误差和最大误差。虽然就这样也没有问题，但每次都要写上“误差”很繁琐。像这种情况，可以对表头的项目进行分级。

分级之后的例子如表**??**所示。在这里通过添加了一条横线，实现了对原本独立各列的整合。这里写着“误差”的部分跨越了多个列，可用`\multicolumn`命令来实现。此外，“局部横线”可以通过`\cmidrule`命令来实现。

方法	最小误差	平均误差	最大误差
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 16: 原本的表格

	误差		
方法	最小	平均	最大
Isomap	0.23	0.44	0.92
LLE	0.10	0.73	1.82

表 17: 将列进行分级后的例子

“列的分级”同样也可以进行多次，其例子如表**??**所示。这里表示了各个县的各种信息。将气温和降水量进行分级后，可以使表格变得更加易于观看。同时，由于将单位的声明都写到了上面，所以使表格在横向上缩短了很多，使得没有分级的项目（县名和人口密度）也能被放到同一个表格里了。

	气温（°C）			降水量（mm）		交通方式		
县名	最高	平均	最低	8 月	12 月	新干线	飞机	人口密度（人/km^2）
石川县	32	20	-1	179.8	304.7	✓	✓	267
静冈县	27	23	5	250.9	63.0	✓	✓	461
冲绳县	33	28	15	175.4	104.4		✓	643

表 18: 对列进行多次分级的表格

3.5 局部横线：cmidrule

在这里来介绍一下在“列的分级”里提过的\cmidrule吧。表??的代码如下所示。

```
\begin{tabular}{@{}llll@{}} \toprule
          & \multicolumn{3}{c}{误差} \\ \cmidrule(l){2-4}
   方法    & 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}中的“l”指的是是否删掉横线的两端。l 表示去掉左侧，r 表示去掉右侧，lr 表示将两端都去掉。通过这种细微的删减，来调整表格的外观。具体例子如表??到表??所示。虽然只有细小的差别，但知道的话会很方便。{2-4} 指的是对应的列数。通过不断改变设置，来对表格进行调整就可以。

	误差		
方法	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}

	误差		
方法	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}

	误差		
方法	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}

	误差		
方法	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}

顺便一提，表??是将\cmidurle设置成了“\cmidrule(lr){2-4} \cmidrule(lr){5-6} \cmidrule(lr){7-8}”，加了“lr”去掉了线的两端。如果不去掉两端的话就如表??所示。这里删除了所有的“lr”，设置成了\cmidrule(){2-4}的形式。这样导致“气温”“降水量”“交通方式”之间的线都连到了一起，因此这里加入适当的空白会使表格更加便于理解。

	气温（°C）			降水量（mm）		交通方式		
县名	最高	平均	最低	8 月	12 月	新干线	飞机	人口密度（人/km^2）
石川县	32	20	-1	179.8	304.7	✓	✓	267
静冈县	27	23	5	250.9	63.0	✓	✓	461
冲绳县	33	28	15	175.4	104.4		✓	643

表 23: 没在表??中设定 cmidrule 删减的结果

3.6 将列移到行

有时将列的内容移到行会使表格变得更简洁。虽然我**不推荐**这种技巧，但是在实际中经常会遇到这种情况所以还是来介绍一下。举个例子，表??是原本的表格。这里在 3 个数据集上对 2 种方法进行了评价，并列出了运行时间。每行都有一个相应的结果，是一个优秀的表格。对每个方法进行了行的分组。本来这样就可以，但是可能会有以下两个需求。

- 想在同一个 Dataset 里比较不同方法，但是数值不在一起。比如 MNIST 的 k-means (10.2) 和 Ours (8.3) 离得较远。
- 继续增加 Method 或 Dataset 的话，会使表格变得过于竖长。想在横向上增加长度。

遇到这种情况时该怎么办才好呢?

可以考虑**将某一列（在这里是 Dataset）移到行**。“一条数据（一行）”不再是“方法 · 数据集 · 运行时间”，而是变成“方法 · 数据集 1 的运行时间 · 数据集 2 的运行时间 · 数据集 3 的运行时间”。这样转换之后的表格就如表??所示。这种方式同时满足了“将同一个数据集的数值放到相近位置”“增加横向长度”的两个要求。但相应地，表头的部分明显变得冗长了。

面对这种情况，使用“列的分级”后就变成了表??。明显变得好多了，就这样结束也没有问题。只不过这里“Dataset=”在表头出现了 3 次，显得冗余。表??和表??是将表格变得更加简洁的 2 个方法。

表??去掉了表??中“Dataset=”的部分，使表格变得更加容易观看了。在这个例子里，就这样也能让人明白表格的含义。然而，由于完全没写“Dataset=”的说明，所以读者可能会不明白“MNIST”的部分是什么意思。因此，不另外说明 Runtime 下面项目的含义的话，这种写法是行不通的。

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: 原本的表格

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: 将 Dataset 这一列移到行之后。表头变得冗余。

	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: 将列进行分级后的结果。虽然比之前变好了，但“Dataset=”的部分还是冗余。

表**??**把表**??**中的“Runtmie (ms)”改成了“Dataset”，这是最简洁的写法。由于没有解释表格里数值的含义，所以有必要在注释里加上“这是 Runtime (ms) 的表格”的说明。这就是“当表格由唯一一种数据构成时，有时可以通过把对数值的说明放到注释里的方式来减少表头信息量，使表格变得简洁”的方法。顺便一提，这种只含唯一一种数据的表格，与本资料的其他表格相比有一个特点。那就是，表头里不含对元素的说明。我个人不是很喜欢这种方式，我还是认为表头里包含元素说明比较好。不过也可能存在由于空间限制必须要这么写的情况。

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

表 27: 省略了“Dataset=”的情况。虽然变简洁了，但是哪里都没有关于“Dataset=”的信息。

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

表 28: 省略了“Runtime (ms)”的情况。虽然最简洁，但是有必要在注释里加上“这是 Runtime (ms) 的表格”的说明。

回归正题，这次讨论了很多种表格的写法，是因为我们既想从 Method 角度也想从 Dataset 角度进行分析。由于表格只由行和列构成，所以想要在横向和纵向上都进行分析的时候，就会变成表**??**和表**??**这样类似于矩阵的形式。因此，这样就逐渐远离了“一行是一次观测”的原则。这种情况下，可以考虑使用图表。将同样内容用图表形式来表现的话就如图**??**所示。这样既能实现“某个数据集下不同方法比较”也能实现“不同数据集下某个方法比较”。在这次的例子中，就算再增加数据集或方法，图表所占的空间也不会改变。此外，还可以增加误差条或者转换成箱线图的形式。另一方面，使用图表的缺点是无法显示具体的数值。比如当需要和既存方法的数值进行比较时，使用图表就会显得不够方便。建议通过考虑表格和图表的优缺点，来选择最有效的表示信息的方式。

3.7 对表格制作感到无从下手的话

如果你对于表格制作感到无从下手的话，可以试着按照以下顺序来做。

1. 根据“一行是一条数据”的原则来分解表格。就算横向上很短也没关系。
2. 对最重要的那一项反复进行“行的分组”。需要的话也可以进行“行的分级”。
3. 需要的话进行“列的分级”。
4. 如果还是感到不合适，可以进行“将列移到行”和“行的分级”。然后省略掉冗长的说明。
5. 如果还是感到不合适，可以考虑使用图表。

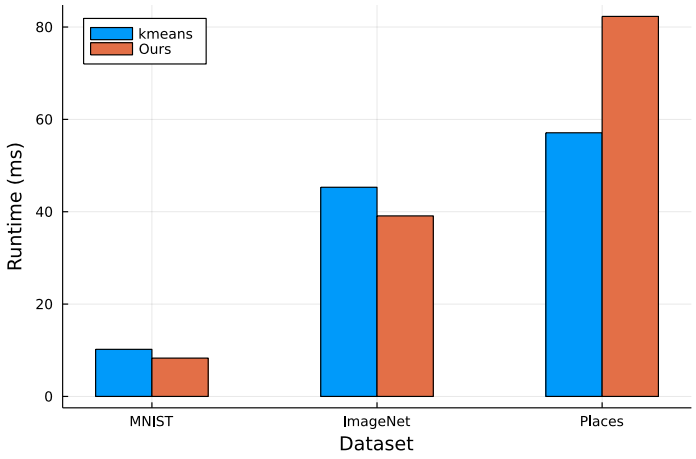

图 1: 把表??和表??转换成图表。

4 案例分析

接下来通过几个例子，来学习怎样能使表格变美观吧。首先请看表??、表??和表??。这里显示的是改变参数时各个方法的精度。看上去它们全都反映了同样的内容，但其实这里面包含了常见的错误。

首先来看表??。它表示的是当改变参数 T 时，ResNet50 和我们的方法的精度变化。这个表格第一眼看上去好像很工整，但是仔细看就会发现，本应是表头的第一行里，并不完全是表头信息。比如 ResNet50 的上面本来应该是“Method”，然而这里却变成了 T。此外，“0.32”上面的“1”也不能成为对数据的说明。因此，这不是一个由“表头 + 行的集合”所构成的表格。另外，这个表格里也没有对要素的说明（即说明数据表示的是精度）。也就是说，不明白数字的含义，必须要添加注释才行。对于这种小表格而言，不用注释就能在表中把所有信息都展示出来的表格应该是最整洁的。

相信很多人会把同样的内容用表??这种形式来表示。左上角的格子里包含“斜线”和“横向纵向的信息”。由于在小学和中学学过这种方法，所以相信很多人都会这么写。然而这种方式也不满足“表头 + 行的集合”的形式。此外，这种写法经常需要添加竖线。

可能还有人会写成表??的形式。它比表??稍好，也能让人明白“0.32”上面是“$T = 1$”。然而还是不能让人明白数字的含义是什么。

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

表 29: 常见例子 1

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

表 30: 常见例子 2

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

表 31: 常见例子 3

那么，怎样才能将这些表格变好看呢？首先试着将表格分解至“一行是一条数据”。这里所谓的“一条数据”指的是“使用某个方法和某个参数 T 时，测量结果的精度（mAP）”。结果如表??所示。这里在表头注明了各个要素是“精度”。虽然这样就可以了，但还是再来进一步思考一下吧。实际上这个表格和表??的结构是相同的。因此进行“将 T 这一列移到行”和“列的分级”，整合后得到表??。这就相当于把表??转换成表??的过程。由于这里 $T = 1$ 的描述已经很短了，所以这个表格已经可以最直观地展示信息了。与表??相比，“mAP”的信息出现在了表格里，并且也增加了“Method”的说明。

不过，能将表格进一步省略成表??或表??的形式吗？结果如表??和表??所示。表??里省略了“$T =$”的部分。在这个例子中，这个省略是行不通的，因为会让人不明白 1, 4, 16 是什么。所以还是不要这么写比较好。表??里省略了“mAP”。这种情况下，需要在注释里加上对它的说明。在这个例子里似乎没有做这种省略的必要。

此外，这次的例子还可以用图表来表示，如图??所示。由于在这里 T 是可连续变化的变量、因此使用了折线图而不是柱状图。请根据情况来决定图表和表格哪一个更好。

接下来，再来看一个例子吧。请看表??。这个例子与表??类似，看上去好像很工整，但第一行不能被称之为表头，因此不能被称为一个优秀的表格。并且不同于表??的第一列能用“Method”来概括，这个例子里找不到能够概括的单词。这要怎么办才好呢？其实仔细观察这个例子就会发现，把行和列进行交换后的表??更加整洁，符合“每一行是一条数据”的原则。因此，从以行为本的角度来说，表??是更加优秀的表格。然而，这里表头的项目太长，使得表格里存在较多空白。怎样才能有效利用空白就需要各自的技术了。比如可以把单位放到下一行等。

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: 分解至“一行是一条数据”后的结果

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

表 33: 把 T 那一列移到行并进行列的分级后的结果。在这个例子中这种形式是最直观的。

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

表 34: 省略了“$T =$”的情况。这里会让人不明白 1, 4, 16 的含义，不是一个优秀的表格。

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

表 35: 省略了“mAP”的情况。需要在注释里加上“这是 mAP 的表格”。

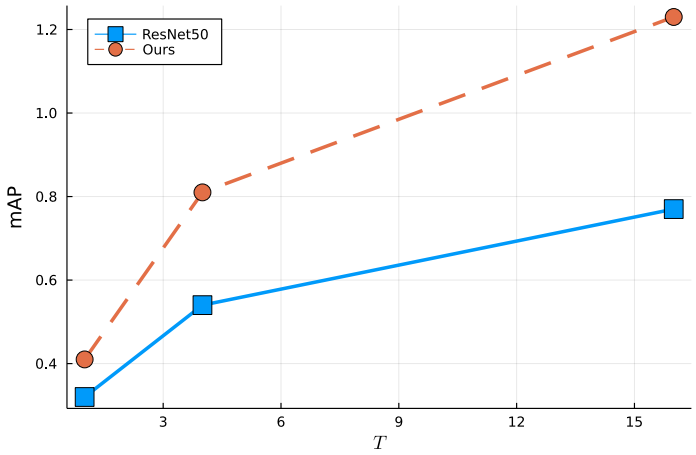

图 2: 把表??转换成图表。

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

表 36: 原本的表格

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

表 37: 把行和列进行交换后的表格