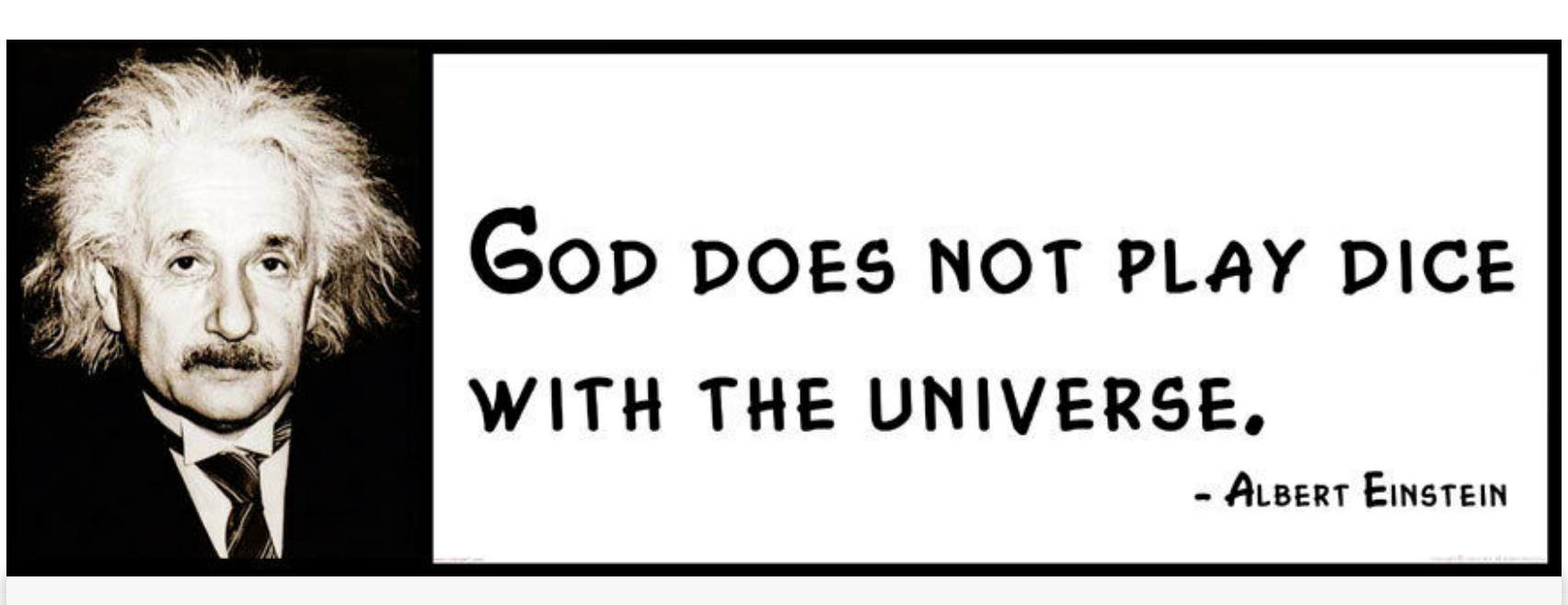
恕我直言，你的实验结论可能严重依赖随机数种子！

原创 python 夕小瑶的卖萌屋 2021-08-19 22:20



文 | python
编 | 小轶

> God does not play dice with the universe ......
> But BERT Does！

包括BERT在内的预训练模型已经是现今NLP工作的标配。但你有没有考虑过，这些工作的实验结论可能都是虚假的？在 Bertology 中，大家从 huggingface 上下载 Google 训好的模型，在精调中结合改进，并应用于下游任务。所有的工作都是基于一组特定的初始化参数，而这个参数严重依赖于预训练时选用的随机数种子（用于参数初始化与预训练数据排序）。这种条件下，你如何知道**你取得的提升，是源于模型方法的改进，还是因为你的方法完美配合了训练BERT时的随机数种子？会不会更换了预训练BERT时的随机数种子，基线方法反而能取得更优秀的表现？**

为了帮助研究者更好地探究这一问题，Google开源了 MultiBERTs， 一组25个不同随机数种子下BERT预训练的结果。外加部分中间结果，一共有165个储存点（checkpoints）。同时，Google提出了Multi-Bootstrap方法，利用不同种子下的BERT预训练结果，检测实验结论是否源于预训练阶段的模型随机性。

> 简便起见，后文将“最初Google开源的BERT版本”称为“Google原版BERT”，以显示特指。

论文题目:
The MultiBERTs: BERT Reproductions for Robustness Analysis

论文链接:
https://arxiv.org/abs/2106.16163

项目地址:
http://goo.gle/multiberts

MultiBERTs

多种子BERT预训练结果

MultiBERTs的本意是提供不同随机数种子下BERT预训练的结果，以供研究者对结论的健壮性与稳定性开展研究。因此，本文作者尽量按照原始BERT训练的参数进行复现，然而作者却无法完美复现BERT论文中的结果，只能尽量去接近。（**Google自己都无法复现BERT的结果，你精调时加的魔改真的靠得住么**？）

具体而言，相对于原版BERT，本文的主要改动为：

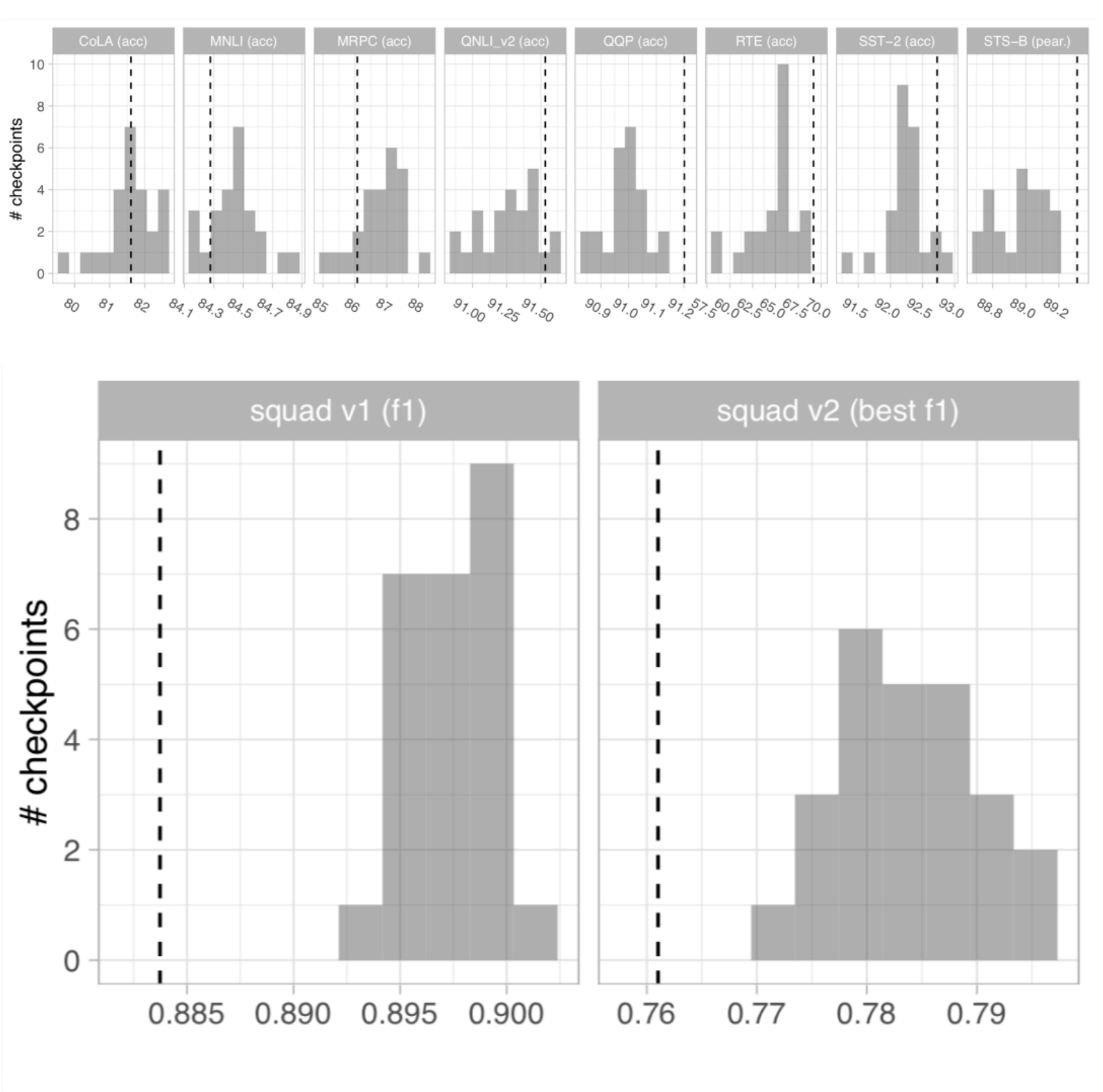
- 训练的步数为2M步，而非原始BERT中的1M步。
- 保持序列长度为512 节点，掩码预测的内容为80个节点。而非原始BERT中90%的步数输入128的节点长度，10%的步数输入512的节点长度。
- 每次训练均采用不同的随机数种子，用以初始化及对训练数据排序。

> 其它未改动的地方：结构（12层transformers+768隐层节点）、预训练任务（MLM&NSP）、预训练语料（由于BooksCorpus不可获取，本文用的[1]中的版本）、batch size（256）、优化器（Adam，lr=1e-4，10K 热身step）、初始化分布（truncated Normal distribution）

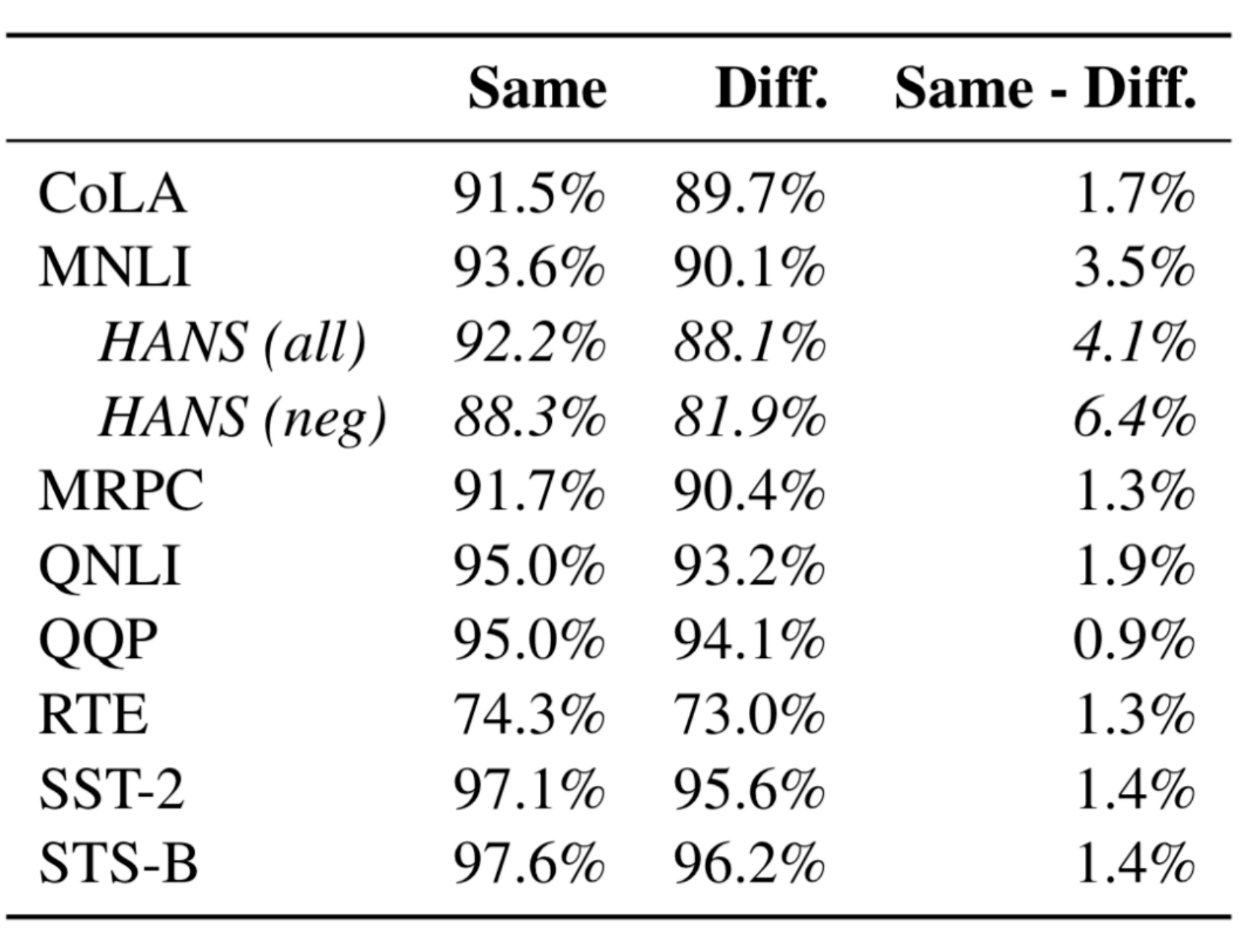
具体的，在GLUE及SQuAD下游任务上的表现如下图所示。每张图为25个格子，分别表示25个不同随机数种子下预训练的模型，在对应任务验证集上的表现。对每个预训练的随机数种子和每个下游任务的组合，均采用五次实验计算均值的方式汇报。虚线表示原始BERT汇报的结果。

作者表示，只优化1M步的话GLUE上的表现比不上BERT，然而2M步的话，GLUE上表现没问题了，但SQuAD上表现又比BERT高了。所以就定成这样了。

> 小编按：与原始BERT相比，作者采用全长（512节点）的预训练输入序列训练了更多步数。经验表明，SQuAD在预训练序列更长时表现更好，因为SQuAD的输入文本相对较长。所以作者相当于完全复现不出原始BERT的效果，转而用更多的步数及更多的全长预训练来弥补表现上的差距。

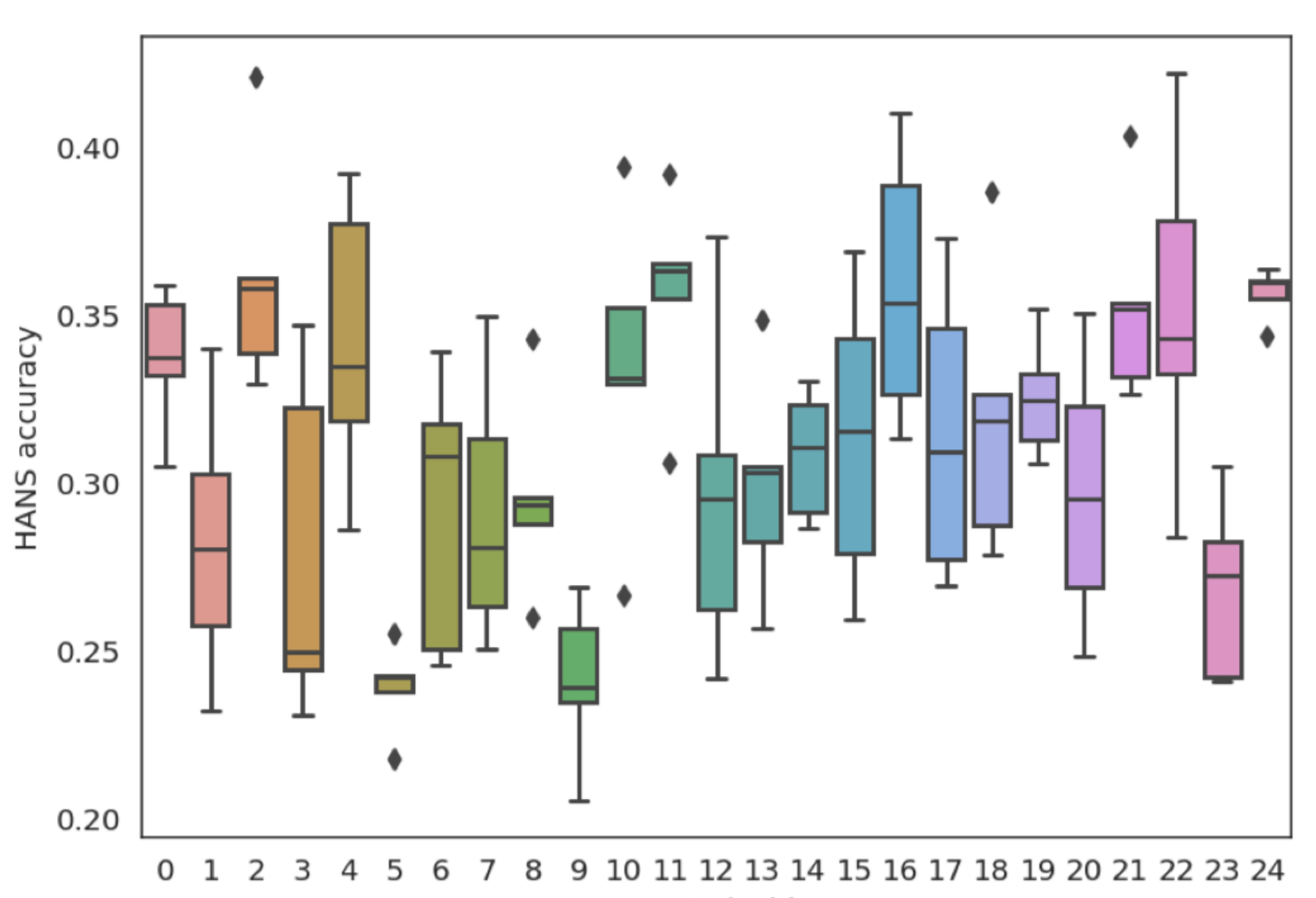


作者表明，在预训练阶段采用不同的随机数种子，对模型表现稳定性有较为明显的影响。如下表所示。这里统计的是下游任务中，样例级别（instance-level）的一致率。**Same**表示采用相同的预训练种子，**Diff**.表示采用不同的预训练种子。从表中可以看出，预训练时使用相同的随机数种子，模型在下游任务中保持有较高的一致率（90%以上）；而如果预训练时采用不同的随机数种子，下游任务的预测一致率会有1%~2%的下降。这一下降在HANS[2]（一个MNLI相关的对抗样本数据集）上表现得更为明显，可以达到 **4%~7%** 。



	Same	Diff.	Same - Diff.
CoLA	91.5%	89.7%	1.7%
MNLI	93.6%	90.1%	3.5%
HANS (all)	*92.2%*	*88.1%*	*4.1%*
HANS (neg)	*88.3%*	*81.9%*	*6.4%*
MRPC	91.7%	90.4%	1.3%
QNLI	95.0%	93.2%	1.9%
QQP	95.0%	94.1%	0.9%
RTE	74.3%	73.0%	1.3%
SST-2	97.1%	95.6%	1.4%
STS-B	97.6%	96.2%	1.4%

特别地，25个预训练随机数种子下，BERT模型在HANS(neg)[2]上的表现，如下图所示。可以看到，随着预训练中随机数种子的变化，模型在**下游任务上的准确率可以有超过20%的波动**。远大于同预训练种子的10%以内的准确率波动。因此，**你的论文的实验结论可能仅在一个BERT的随机种子下成立。更换预训练种子之后，结论可能不再成立。**我们也可以由此看到，利用不同随机数种子下BERT预训练的结果，对探究实验结论的鲁棒性十分必要。



Multi-Bootstrap

基于多种子结果的自举评估方法

有了不同的种子下训练的BERT结果：**MultiBERTs**，作者进一步提出了 **Multi-Bootstrap** 方法，以自举（Bootstrap）策略对模型的鲁棒性与结论的随机性做非参数估计。

自举（Bootstrap）听上去很玄幻。据小编的理解，这种方法就是一个对数据进行采样均值的过程。具体而言，我们先从预训练随机数种子中，有放回地采样n_s个种子$(S_1, S_2, \dots, S_{n_s})$。然后，同样有放回地，从测试数据中，采样$n_x$个测试点：$((X_1,Y_1),(X_2,Y_2),\dots,(X_{n_x},Y_{n_x}))$ 记f_s为种子s下的模型映射，l为打分函数，于是模型表现的估计量为：

$$\hat{L}(s) = \frac{1}{n_x}\sum_{i=1}^{n_x} l(f_s(X_i), Y_i)$$
$$\hat{\theta} = \frac{1}{n_s}\sum_{j=1}^{n_s} \hat{L}(S_j)$$

通过多次采样，我们可以得到$\hat{\theta}$的期望及标准差的估计(estimation)，并用这个估计对实验结论的鲁棒性进行评估。

如果评估还涉及到下游任务的随机数种子，在上面的采样均值过程中，再加一层对下游任务的随机数种子采样均值即可。

具体的应用形式，可以分成以下4种：

- 对比基线：将基于MultiBERTs的结果同固定基线进行对比。这里的固定基线可以是随机结果、人类表现、或原版BERT没有对随机数种子做采样的结果等。
- 成对采样：对比同一组预训练结果之下的结果，比如均是基于MultiBERTs，探究添加的某个魔改结构是否有帮助。在这种采样中，对待对比的两个模型的随机数种子部分采用同样的采样策略。
- 不成对采样：一般用于不可成对采样的场景。如对比MultiBERTs及某个类似的“MultiRoBERTas”之间的性能差异。因为两种模型并不共享预训练结果的检查点（checkpoints），采样时分别对两边的种子进行采样。
- 假设检验（P-Values）：可以去计算有多大比例的采样结果，得到的表现估计量$\hat{\theta}$超过基线水平。

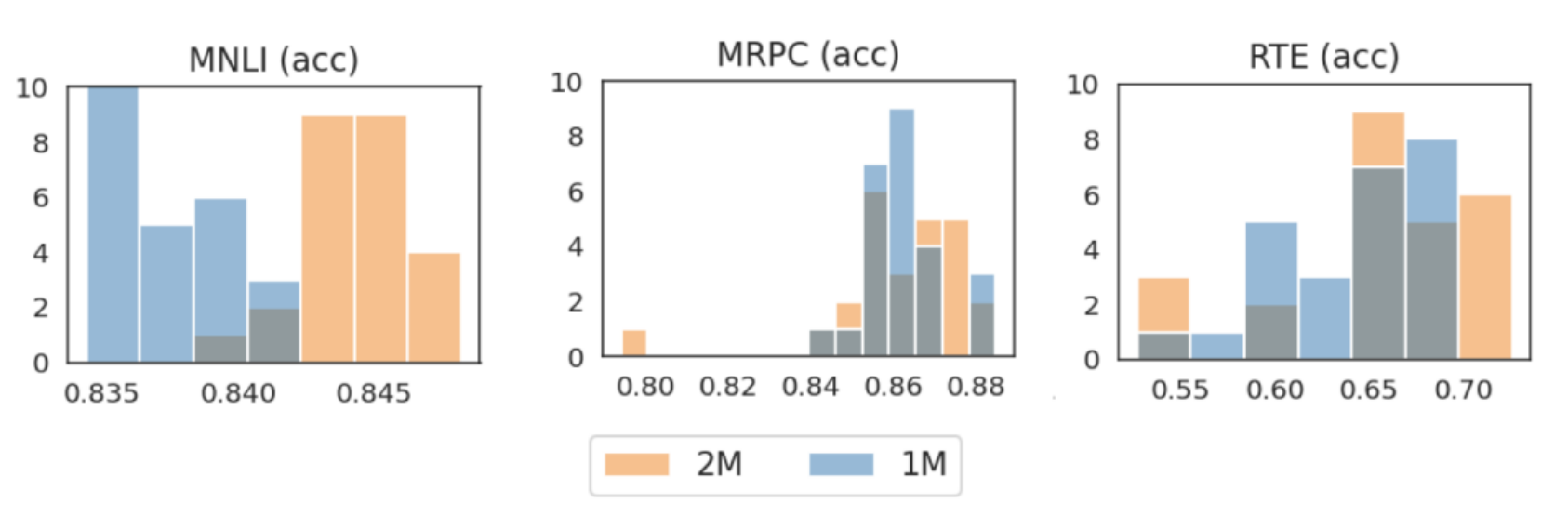
实战示例

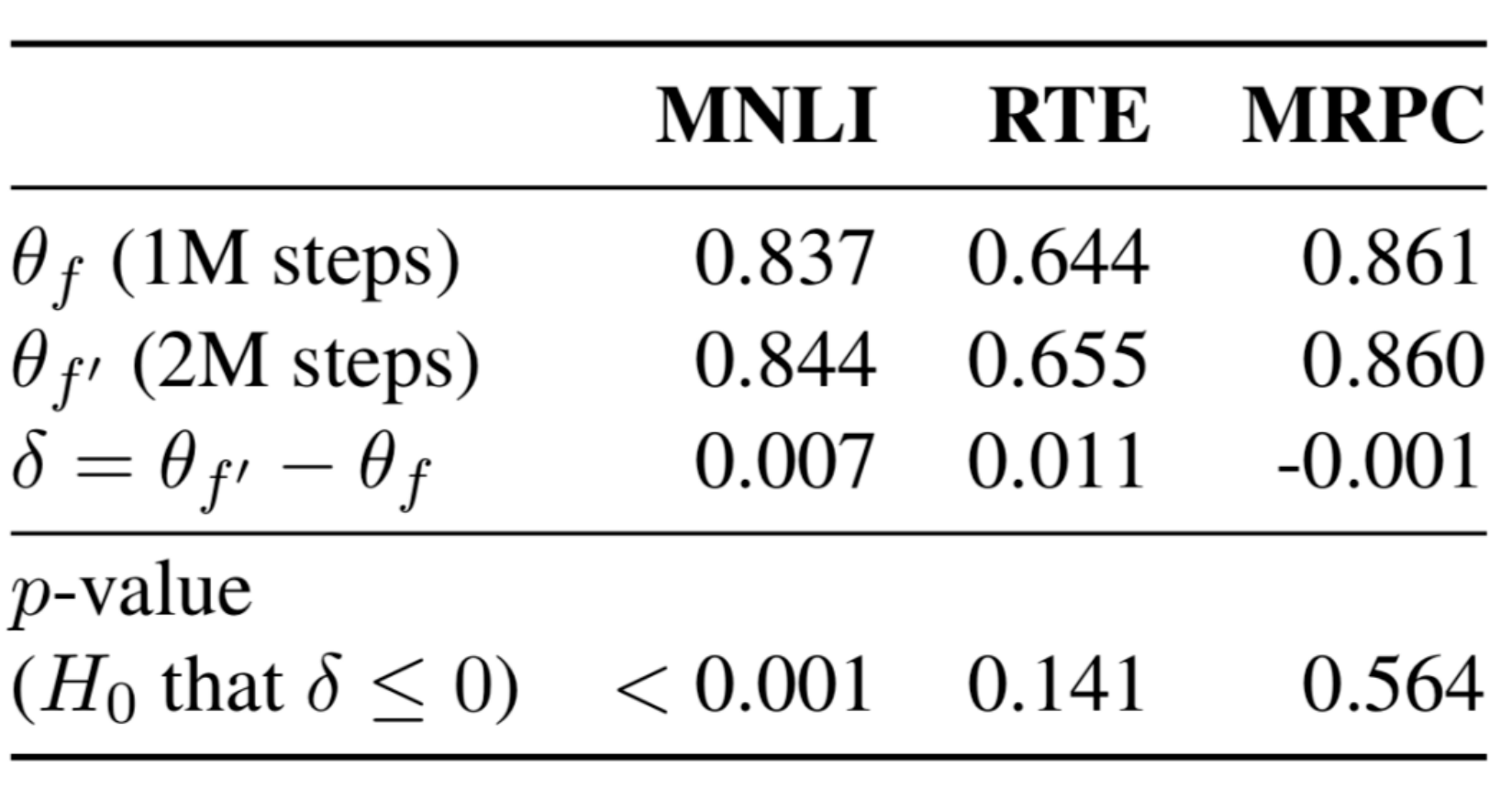
作者在实现MultiBERTs时观察到两个现象：更多的预训练步数普遍带来更好的表现；MultiBERTs 的表现在SQuAD上比原始BERT要好。作者将Multi-Bootstrap应用在对这两个问题的探究上，以体现该方法的有效性。

更多的预训练步数可以取得更好的效果吗？(成对采样)

一般人们认为预训练模型迭代步数越多，模型的表现就越好。作者在训练MultiBERTs时也发现了类似的现象。那这一结论是否具有统计学意义？作者这里采用成对采样策略的Multi-Bootstrap方法，对比迭代2M步和1M步预训练的BERT模型，在下游GLUE任务上的表现。

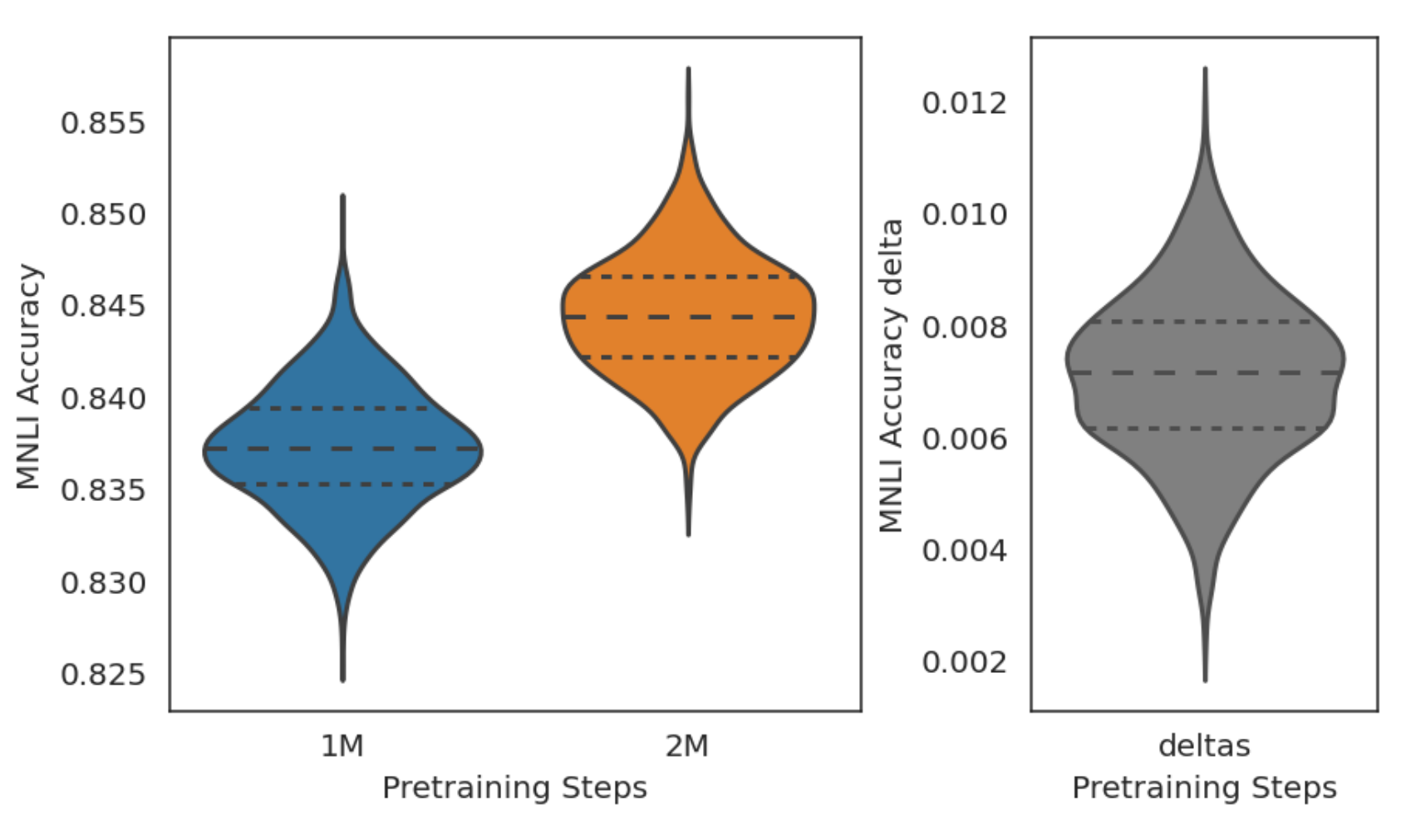
对比如下图，可以看到，对于MNLI任务而言，更多的迭代步数会显著带来性能提升，p-value<0.001。而对于MRPC、RTE等任务而言，更多的预训练步数对下游任务的提升就值得怀疑了，p-value分别只有0.564和0.141。





	MNLI	RTE	MRPC
θ_f (1M steps)	0.837	0.644	0.861
$\theta_{f'}$ (2M steps)	0.844	0.655	0.860
$\delta = \theta_{f'} - \theta_f$	0.007	0.011	-0.001
p-value (H_0 that $\delta \leq 0$)	< 0.001	0.141	0.564

利用成对采样的策略可以进一步看出，虽然MNLI任务上，1M和2M迭代步数的预训练模型性能分布有较为明显的重叠。但两者具有较为明显的相关性，即在同一随机种子下，2M迭代的模型表现有很大概率高于1M迭代的模型。这导致了极高的显著性。



MultiBERTs 的表现在SQuAD上比原始BERT要好？(对比基线)

类似地，作者也对比了MultiBERTs和原版BERT在SQuAD2.0任务上的性能差异。结果表明，MultiBERTs性能超过原版BERT的p-value<0.001，具有极高的显著性。

因为原版BERT没有提供随机种子，所以作者建议在这种模型下，同时汇报性能差异的95%置信区间。MultiBERTs比原版BERT在SQuAD2.0任务上的性能提升量的95%置信区间为提升1.9%~2.9%。

开放问题

论文作者指出，有了MultiBERTs，研究者还可以在替换预训练过程中的随机数种子的前提下，进一步探索下面这些结论：

- 是否仅有Google原版BERT能编码句法信息、世界知识？
- 是否仅有Google原版BERT包含了社会偏见(social stereotypes)?
- RoBERTa等模型，是否只超过了Google原版的BERT？
- 引入NLI等中间训练任务，是否可以对下游任务带来稳定提升？
- 减少attention头的数量，是否稳定影响下游任务表现？
- BERT中引入语义角色信息是否可以稳定提升下游任务效果？

坑挖好了，工具也有了，问题也提出了，小伙伴们还等什么？还不赶紧把代码跑上，灌上一波？(ง •̀_•́)ง

参考文献

[1] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. arXiv preprint arXiv:1908.08962.

[2] Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In Proceed- ings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3428–3448, Florence, Italy. Association for Computational Lin- guistics.