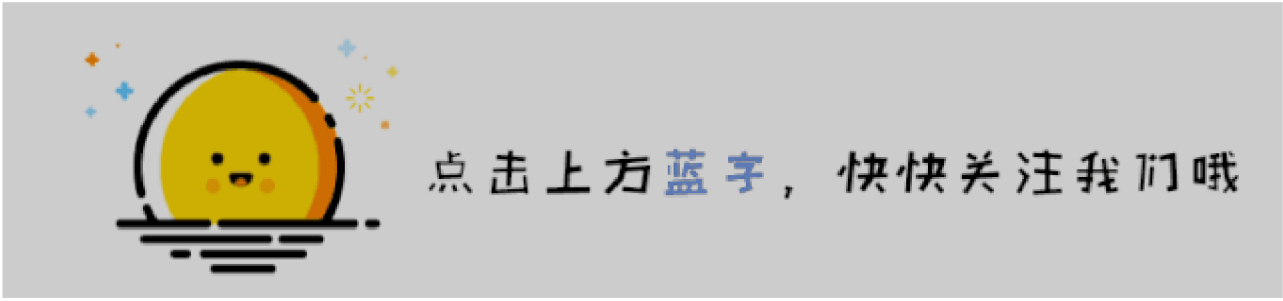
[预训练语言模型专题] MT-DNN(KD)：预训练、多任务、知识蒸馏的结合

原创 管扬 朴素人工智能

来自专辑

预训练语言模型



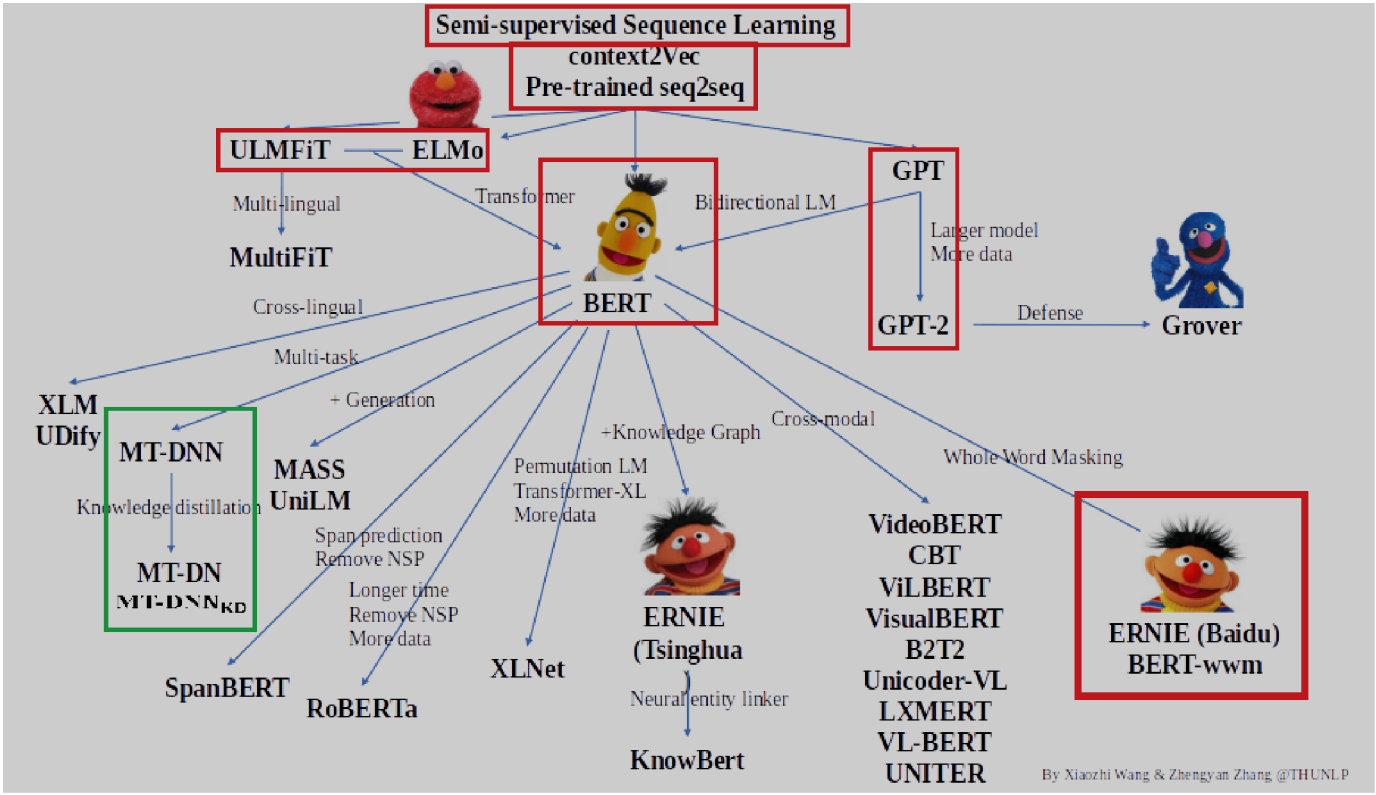
本文为预训练语言模型专题系列第八篇

快速传送门

1-4：[萌芽时代]、[风起云涌]、[文本分类通用技巧] 、 [GPT家族]

5-7：[BERT来临]、[浅析BERT代码]、[ERNIE合集]

感谢清华大学自然语言处理实验室对**预训练语言模型架构**的梳理，我们将沿此脉络前行，探索预训练语言模型的前沿技术，红框中为已介绍的文章，绿框中为本期介绍的文章，欢迎大家留言讨论交流。



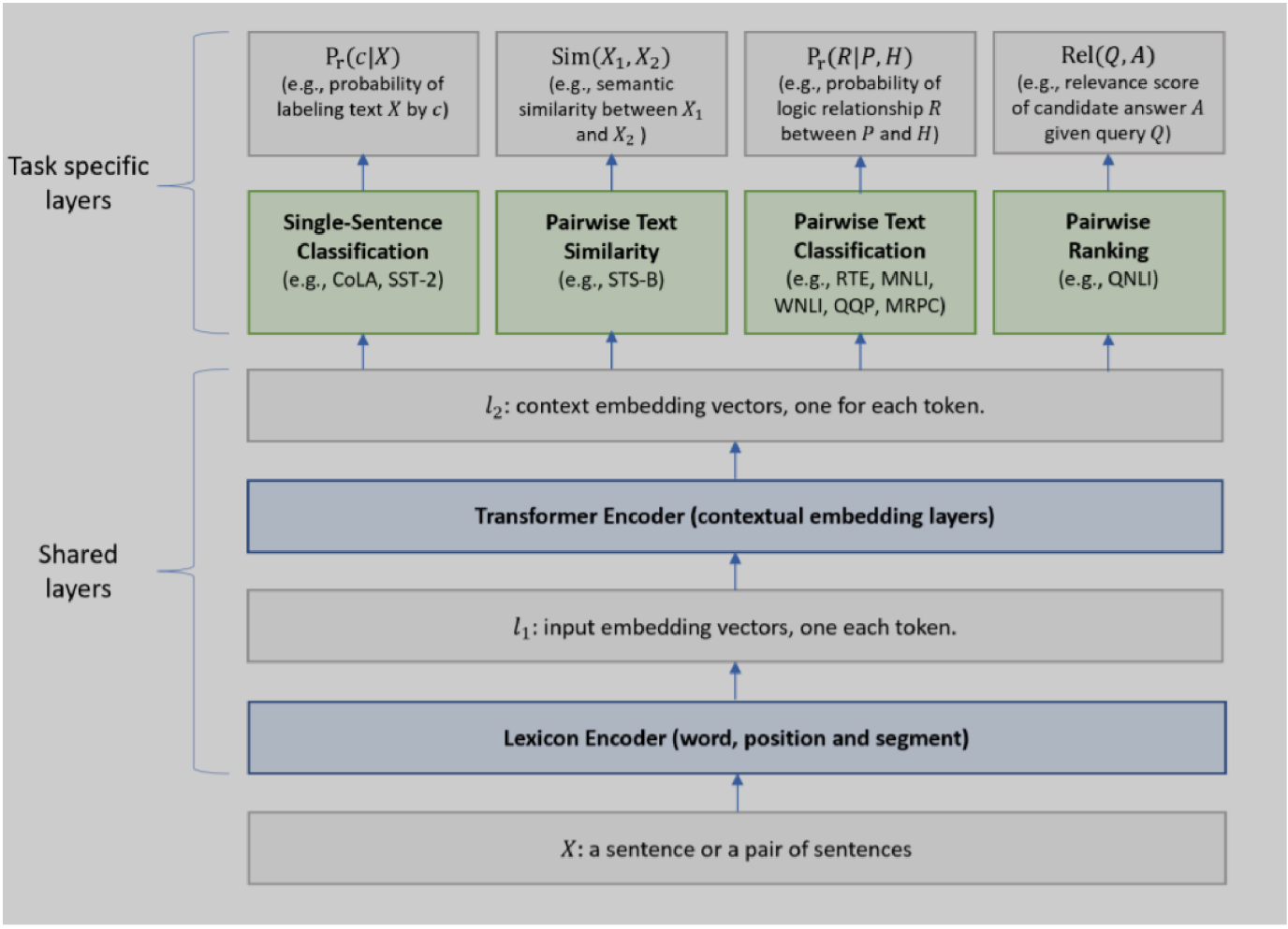
1

Multi-Task Deep Neural Networks for Natural Language Understanding（2019）

本期介绍的是由微软提出的 **Multi-Task Deep Neural Networks**。众所周知，语言模型预训练方法和多任务学习策略都是提高模型性能的重要手段，本文就结合了两者的优点，提出了MT-DNN的方案，并在GLUE上的八个NLU任务上超越了之前的state-of-art模型。

首先，MT-DNN考虑了四种类型的NLU任务，分别是单句文本分类(CoLA, SST-2)，文本对的分类(RTE，MNLI，QQP，MRPC)，文本相似度度量(STS-B)，相关度排序(QNLI)。括号中是在GLUE中四种类型对应的任务。这四种类型的任务在MT-DNN中对应着三种损失函数来优化，分别是**分类，回归和排序** 。

其次，MT-DNN的结构如下图，低层的结构是对所有任务通用的，高层结构则是对应特定的任务。底层的词经过embedding获得词级别的表示l1，再经过Transformers通过self-attention获得上下文语义的表示l2，两者都是会通过多任务来训练出的共同的语义表示。再往上就是对应特定任务的头，经过特定的任务损失函数来进行训练。



MT-DNN数据构造的方式和BERT差不多，开头[CLS]，两句句子中用[SEP]隔开，结尾[SEP]。上面四种类型任务的数据构造都可以遵循这种方式。

MT-DNN的训练也分成两个阶段，预训练和多任务学习。预训练的任务与BERT一致，有MLM和NSP，这里不再赘述。整体的训练方式我们可以看下图的流程。首先，对包括Transformer encoder 在内的模型中的**共享层**进行预训练。接着，将数据混合起来，每次取一个小的batch，它会是属于某个任务T的。如果这个任务T是分类任务，则用分类任务的公式计算loss，如果是回归任务或排序任务，则用回归任务或排序任务的公式计算loss，接着利用loss更新梯度。

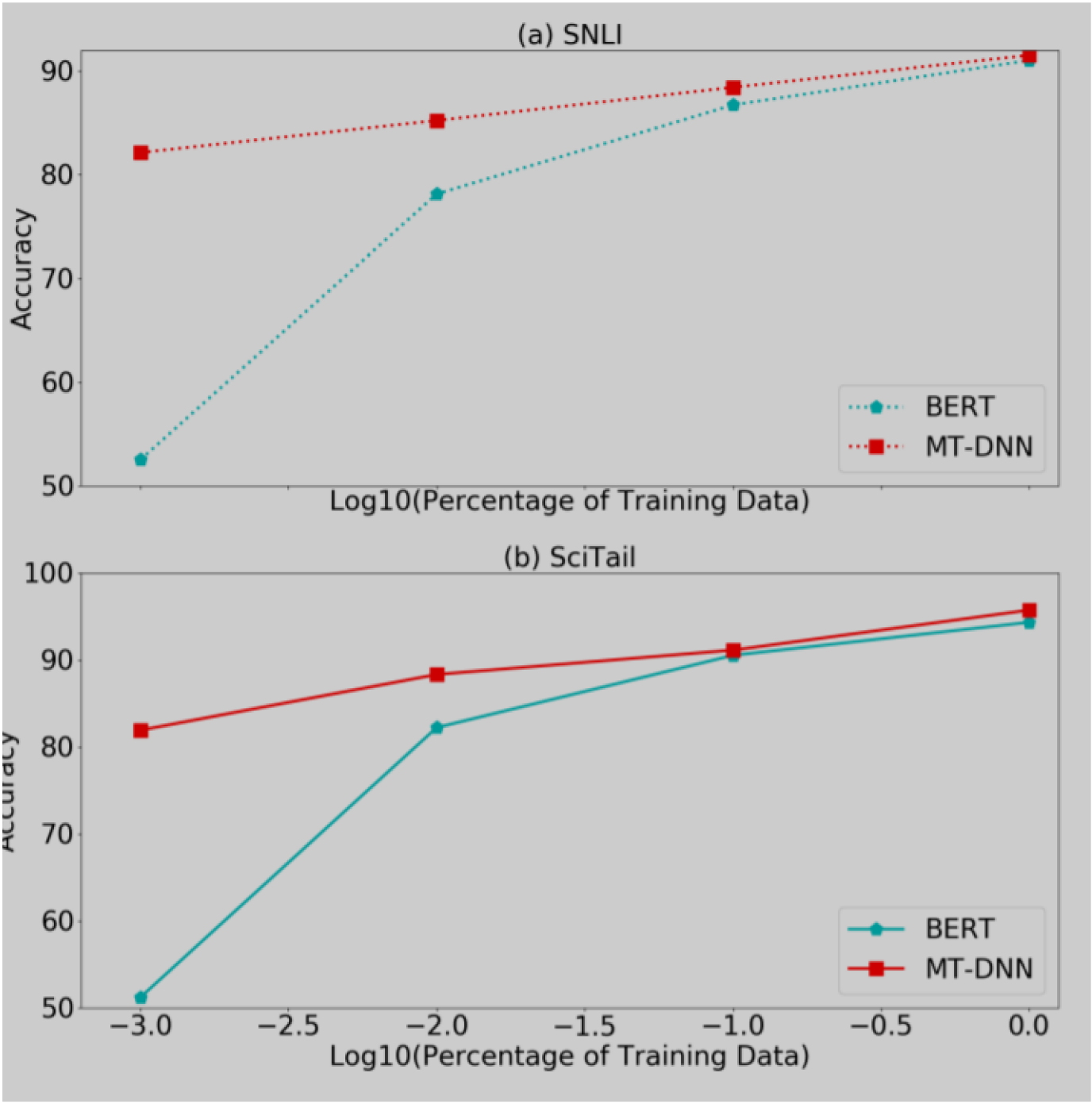
Algorithm 1: Training a MT-DNN model.

Initialize model parameters Θ randomly.
Pre-train the shared layers (i.e., the lexicon encoder and the transformer encoder).
Set the max number of epoch: $epoch_{max}$.
//Prepare the data for T tasks.
for t *in* $1, 2, ..., T$ **do**
  Pack the dataset t into mini-batch: D_t.
end
for $epoch$ *in* $1, 2, ..., epoch_{max}$ **do**
  1. Merge all the datasets:
    $D = D_1 \cup D_2 ... \cup D_T$
  2. Shuffle D
  for b_t *in* D **do**
    //b_t is a mini-batch of task t.
    3. Compute loss : $L(\Theta)$
      $L(\Theta)$ = Eq. 6 for classification
      $L(\Theta)$ = Eq. 7 for regression
      $L(\Theta)$ = Eq. 8 for ranking
    4. Compute gradient: $\nabla(\Theta)$
    5. Update model: $\Theta = \Theta - \epsilon\nabla(\Theta)$
  end
end

实现细节上，MT-DNN是基于BERT的pytorch实现，优化器使用Adamax，学习率5e-5，batch_size32，总共训练五轮。使用了linear decay，并且加上了0.1的warmup。使用了大约为0.1的dropout，并且将梯度剪切到1，接下来，我们看看模型的结果，排行榜信息截至2019年2月25日。

Model	CoLA 8.5k	SST-2 67k	MRPC 3.7k	STS-B 7k	QQP 364k	MNLI-m/mm 393k	QNLI 108k	RTE 2.5k	WNLI 634	AX	Score
BiLSTM+ELMo+Attn [1]	36.0	90.4	84.9/77.9	75.1/73.3	64.8/84.7	76.4/76.1	-	56.8	65.1	26.5	70.5
Singletask Pretrain Transformer [2]	45.4	91.3	82.3/75.7	82.0/80.0	70.3/88.5	82.1/81.4	-	56.0	53.4	29.8	72.8
GPT on STILTs [3]	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.8/80.6	-	69.1	65.1	29.4	76.9
$\text{BERT}_{\text{LARGE}}$ [4]	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	86.7/85.9	92.7	70.1	65.1	39.6	80.5
$\text{MT-DNN}_{\text{no-fine-tune}}$	58.9	94.6	**90.1/86.4**	89.5/88.8	**72.7/89.6**	86.5/85.8	**93.1**	79.1	65.1	39.4	81.7
MT-DNN	**62.5**	**95.6**	**91.1/88.2**	**89.5/88.8**	**72.7/89.6**	**86.7/86.0**	**93.1**	**81.4**	65.1	**40.3**	**82.7**
Human Performance	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0/92.8	91.2	93.6	95.9	-	87.1

可以看到，MT-DNN效果是要强于BERT的，对特定任务进行finetune会效果更好。上面BERT模型的结果是经过finetune的。另外一点值得注意的是，因为MT-DNN多任务预训练的效果，所以它在迁移的场景中，特别是**数据量小**的情况下，表现更优于BERT，如下图。可以预想到的是，MT-DNN将会更容易适应新的环境和任务。



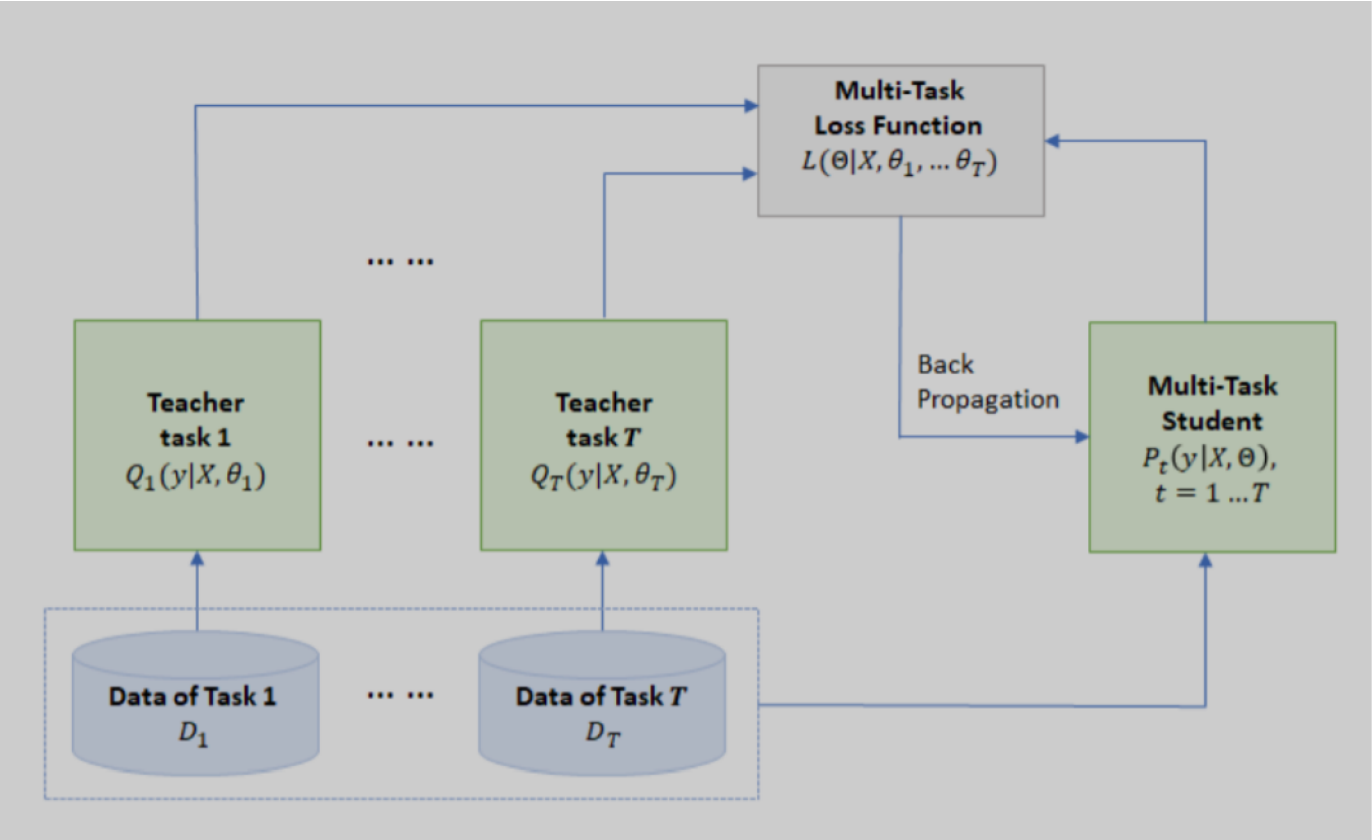
总结一下，MT-DNN基本上还是一个在BERT上的改进，改进的内容是使用了一种**多任务**的训练方式，使模型更加稳定，泛化性更好，且能在更少的数据或者新任务上获得良好的效果。其实我个人挺受这篇文章启发的，因为遵循MT-DNN的思路，其实我们可以让**BERT的预训练**变得更好，使用更多更优秀的任务去进行预训练，甚至结合有标注的数据，这样可以更大程度地利用数据，加强模型的效果。

2

Improving Multi-Task Deep Neural Networks via Knowledge Distillation for Natural Language Understanding（2019）

这篇文章是上一篇文章的延续，主要改进部分是加入了**知识蒸馏**，被称为MT-DNN(KD)，文章中提到，对每个任务，他们训练了多个模型集成而成的MT-DNN作为Teacher Model，它是会强过单个模型的。然后通过多任务学习，将这些Teacher Model的知识蒸馏到单个MT-DNN上，他们观察这样做以后，效果在GLUE的七个任务上大幅提高了。

模型的**集成**已经成为各大排行榜上提高模型成绩的通用做法了。但是集成的模型通常是由许多模型堆叠而成，对资源的要求较高，更别说目前很多模型本身就很大，给在线上的部署带来了很大的挑战。所以本文选择首先对每个集成模型，获得比较强的Teather model，然后通过知识蒸馏的方式，离线地对每个任务生成**软标签**，比如对分类任务来讲就是分类的概率。接着，结合硬标签和软标签，通过多任务学习，训练出一个MT-DNN(Student)，通过这种方式，在不增加系统负担的情况下，把集成的能力带给了学生模型，提供了良好的泛化性和效果。



Hinton2015年的文章曾经说到，软标签中包含了Teacher model是如何泛化的信息，是Student model 获取其泛化能力的关键。在本篇文章中，每一个特定任务的Teacher，都是集成后的MT-DNN，它拥有很好的泛化能力，通过软标签和多任务学习，就能将这个能力传递给Student，让它表现更好。

实现细节上，对每个任务，首先在{0,1, 0.2, 0.3}中选择并改变dropout，训练了六个MT-DNN模型，挑选了在MNLI和RTE上效果最好的三个，接下来把他们在特定任务MNLI，QQP，RTE，QNLI上finetune，并生成软标签。对单个任务，软标签平均作为这个任务最终的软标签。而硬标签软标签的权重作者进行尝试过，并无明显区别。

Model	CoLA 8.5k	SST-2 67k	MRPC 3.7k	STS-B 7k	QQP 364k	MNLI-m/mm 393k	QNLI 108k	RTE 2.5k	WNLI 634	AX	Score
BiLSTM+ELMo+Attn [1]	36.0	90.4	84.9/77.9	75.1/73.3	64.8/84.7	76.4/76.1	79.8	56.8	65.1	26.5	70.0
Singletask Pretrain Transformer [2]	45.4	91.3	82.3/75.7	82.0/80.0	70.3/88.5	82.1/81.4	87.4	56.0	53.4	29.8	72.8
GPT on STILTs [3]	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.8/80.6	-	69.1	65.1	29.4	76.9
$\text{BERT}_{\text{LARGE}}$ [4]	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	86.7/85.9	92.7	70.1	65.1	39.6	80.5
MT-DNN [5]	61.5	95.6	90.0/86.7	88.3/87.7	72.4/89.6	86.7/86.0	-	75.5	65.1	40.3	82.2
Snorkel MeTaL [6]	63.8	**96.2**	91.5/88.5	**90.1/89.7**	73.1/89.9	87.6/87.2	93.9	80.9	65.1	39.9	83.2
ALICE *	63.5	95.2	**91.8/89.0**	89.8/88.8	**74.0/90.4**	**87.9/87.4**	95.7	80.9	65.1	40.7	83.3
$\text{MT-DNN}_{\text{KD}}$	**65.4**	95.6	91.1/88.2	89.6/89.0	72.7/89.6	87.5/86.7	**96.0**	**85.1**	65.1	**42.8**	**83.7**
Human Performance	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0/92.8	91.2	93.6	95.9	-	87.1

上图为模型效果在GLUE上的汇总，排行榜信息截至于2019年4月1日。首先，从最终Score来看，知识蒸馏的MT-DNN超越了其他的state-of-art，名列第一。其次，第三名的Snorkel MeTaL是一个emsemble的模型，第二名的ALICE当时还未发表，无法得知是否为emsemble。从这个结果看，知识蒸馏给MT-DNN带来了很大的提高，甚至超过了很多集成模型。

Model	MNLI-m/mm	QQP	RTE	QNLI(v2)	MRPC	CoLa	SST-2	STS-B
$BERT_{LARGE}$	86.3/86.2	91.1/88.0	71.1	92.4	89.5/85.8	61.8	93.5	89.6/89.3
MT-DNN	87.1/86.7	**91.9**/89.2	83.4	92.9	91.0/87.5	63.5	**94.3**	90.7/90.6
$MT\text{-}DNN_{KD}$	**87.3/87.3**	**91.9/89.4**	**88.6**	**93.2**	**93.3/90.7**	**64.5**	**94.3**	**91.0/90.8**
MT-DNN-ensemble	88.1/87.9	92.5/90.1	86.7	93.5	*93.4/91.0*	*64.5*	*94.7*	*92.1/91.6*

上图可以看到，知识蒸馏的方法确实给模型带来了很大的收益，只略低集成模型一点点，而且又并没有增加模型的负担。这篇论文本质上是讲知识蒸馏在MT-DNN上的应用。硬标签软标签结合训练的方式，以及伪标签，在很多比赛中已经被大家广泛应用，取得了很不错的效果。总的来说，这两篇文章中所使用的技巧都是比较实用可靠的，如果还没有了解的小伙伴，推荐大家自己去试一试哦~

未完待续

本期的论文就给大家分享到这里，感谢大家的阅读和支持，下期我们会给大家带来其他预训练语言模型的介绍，敬请大家期待！

欢迎关注朴素人工智能，这里有很多最新最热的论文阅读分享，有问题或建议可以在公众号下留言。

往期推荐

- 表格问答1， 表格问答2
- [预训练语言模型专题] 百度出品ERNIE合集
- 叫我如何相信你？聊一聊语言模型的校准
- [预训练语言模型专题] Huggingface简介及BERT代码浅析