INVITED TALK AT

BAAI 智源研究院

预训练语言模型研究进展

RESEARCH PROGRESS ON PRE-TRAINED LANGUAGE MODELS

崔一鸣

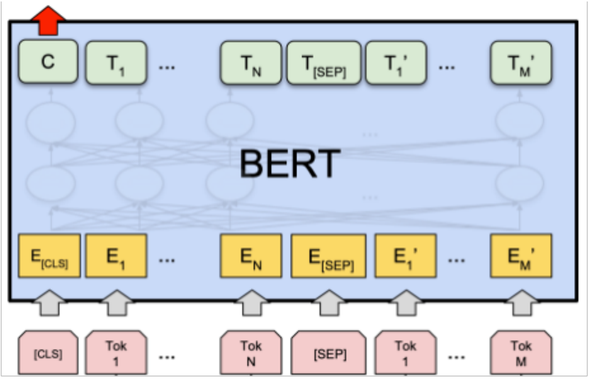
科大讯飞

2022年7月20日

科大讯飞 iFLYTEK

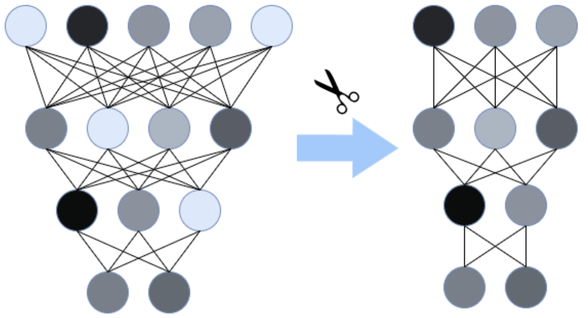
报告内容

- **通用预训练语言模型**
  - 基于全词掩码的中文BERT-wwm
  - 基于纠错型掩码语言模型的MacBERT
  - 基于乱序语言模型的PERT



预训练语言模型

- **预训练语言模型的蒸馏与压缩**
  - 知识蒸馏工具TextBrewer
  - 模型裁剪工具TextPruner



高效训练与推理

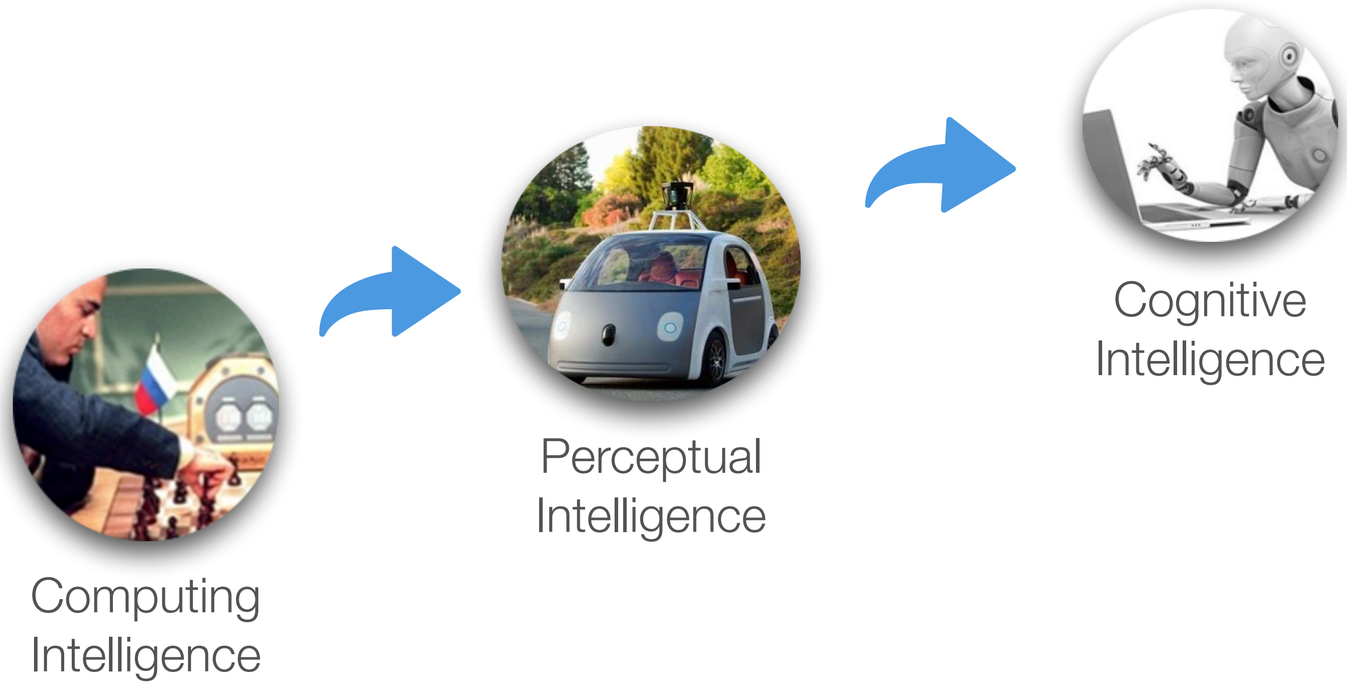
- **总结**

通用预训练语言模型

General Pre-trained Language Model

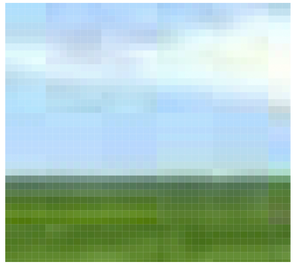
概述

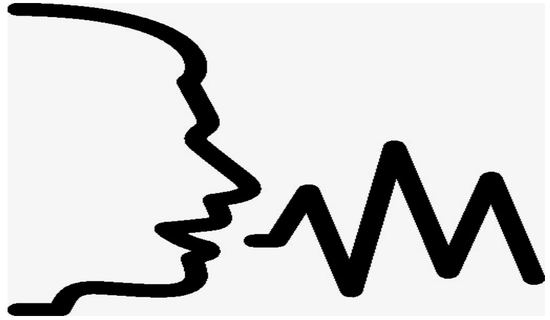
- **人工智能发展的三个阶段**
  - 人工智能的一个重要目标是**让机器能听会说，能理解会思考**
  - 目前人工智能技术正处在从感知智能到认知智能跨越的时间节点
  - 自然语言理解（NLU）是认知智能中的重要内容，是通往强人工智能的必经之路



概述

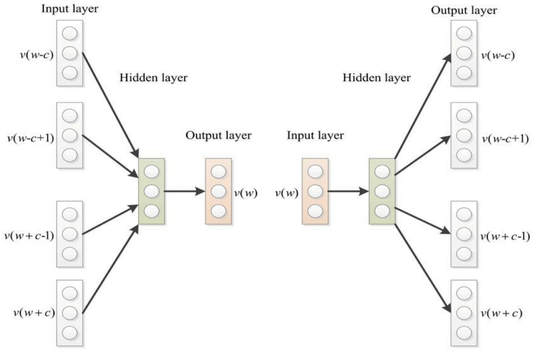
- **自然语言处理为什么难?**
  - 与图像、语音不同的是**语言是高度抽象的产物**，其基本组成单位并不是明确的物理量





概述

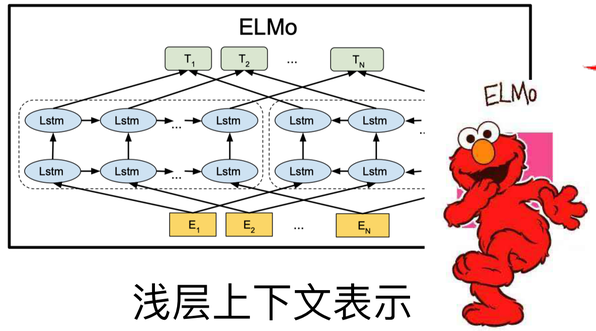
- **自然语言表示的发展一定程度上反映了自然语言处理的发展**
  - 自然语言表示的变迁很大程度影响着自然语言处理的范式
  - 从离散到连续，从上下文无关到上下文相关，从浅层到深层



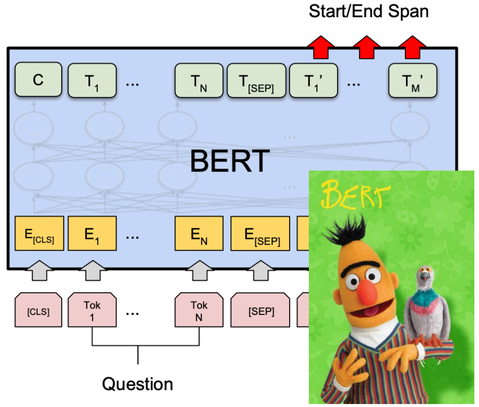
词向量表示

[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

One-hot表示



浅层上下文表示

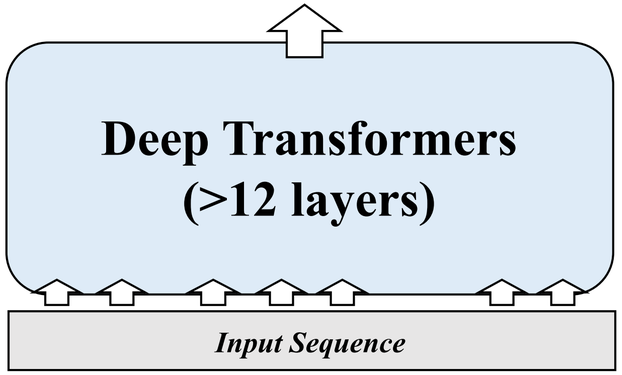


深层上下文表示

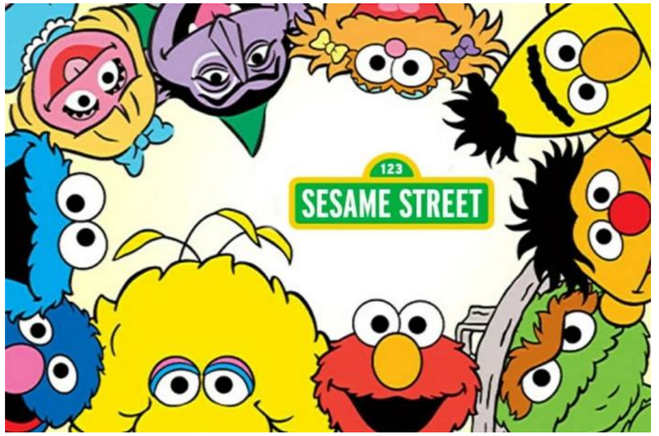
预训练模型三要素

大数据
（无标注文本）

大模型
（深度神经网络）



大算力
（并行计算集群）



GPT

OpenAI

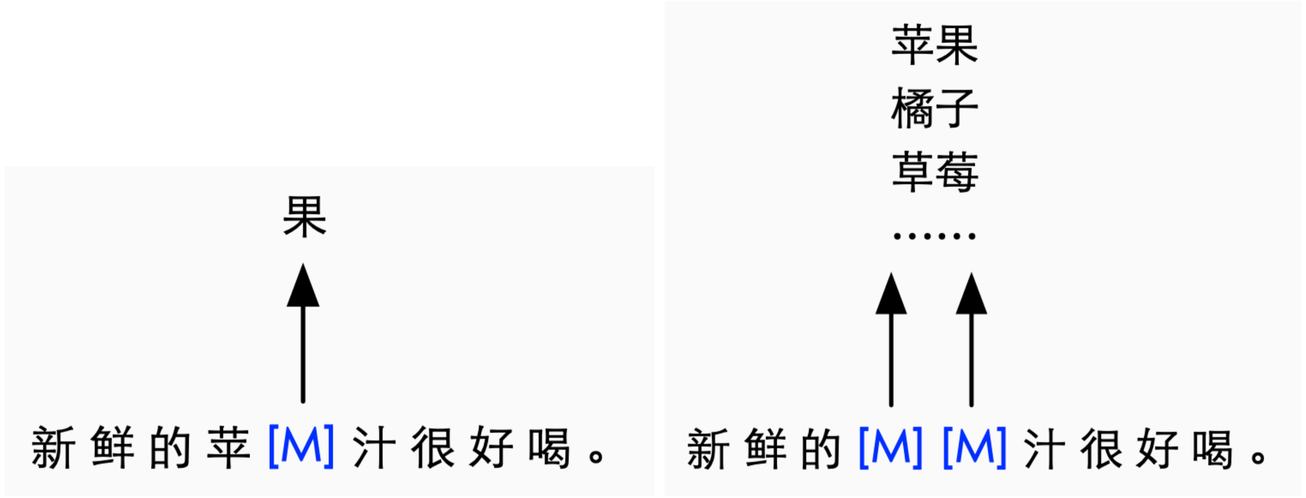
GPT-2

GPT-3

以及更多钱

通用预训练语言模型：中文BERT-wwm

- **基于全词掩码的中文BERT系列模型**
  - 将全词掩码技术（Whole Word Masking，wwm）应用在中文BERT的建模中
  - 不同掩码策略
    - 标准掩码语言模型（MLM）：将输入文本中的字看作独立分布，随机对字进行掩码
    - 全词掩码语言模型（WWM）：考虑了中文分词，将属于同一个词中的所有字进行掩码



(a) 以字为掩码单位　　(b) 以词为掩码单位

[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：中文BERT-wwm

- **实验结果**
  - 基于全词掩码的方法能够显著提升预训练模型性能

Model	SQuAD 1.1	MNLI
BERT-Large, Uncased (Original)	91.0 / 84.3	86.05
BERT-Large, Uncased (WWM)	92.8 / 86.7	87.07
BERT-Large, Cased (Original)	91.5 / 84.8	86.09
BERT-Large, Cased (WWM)	92.9 / 86.7	86.46

▲ 英文任务效果（from Google）

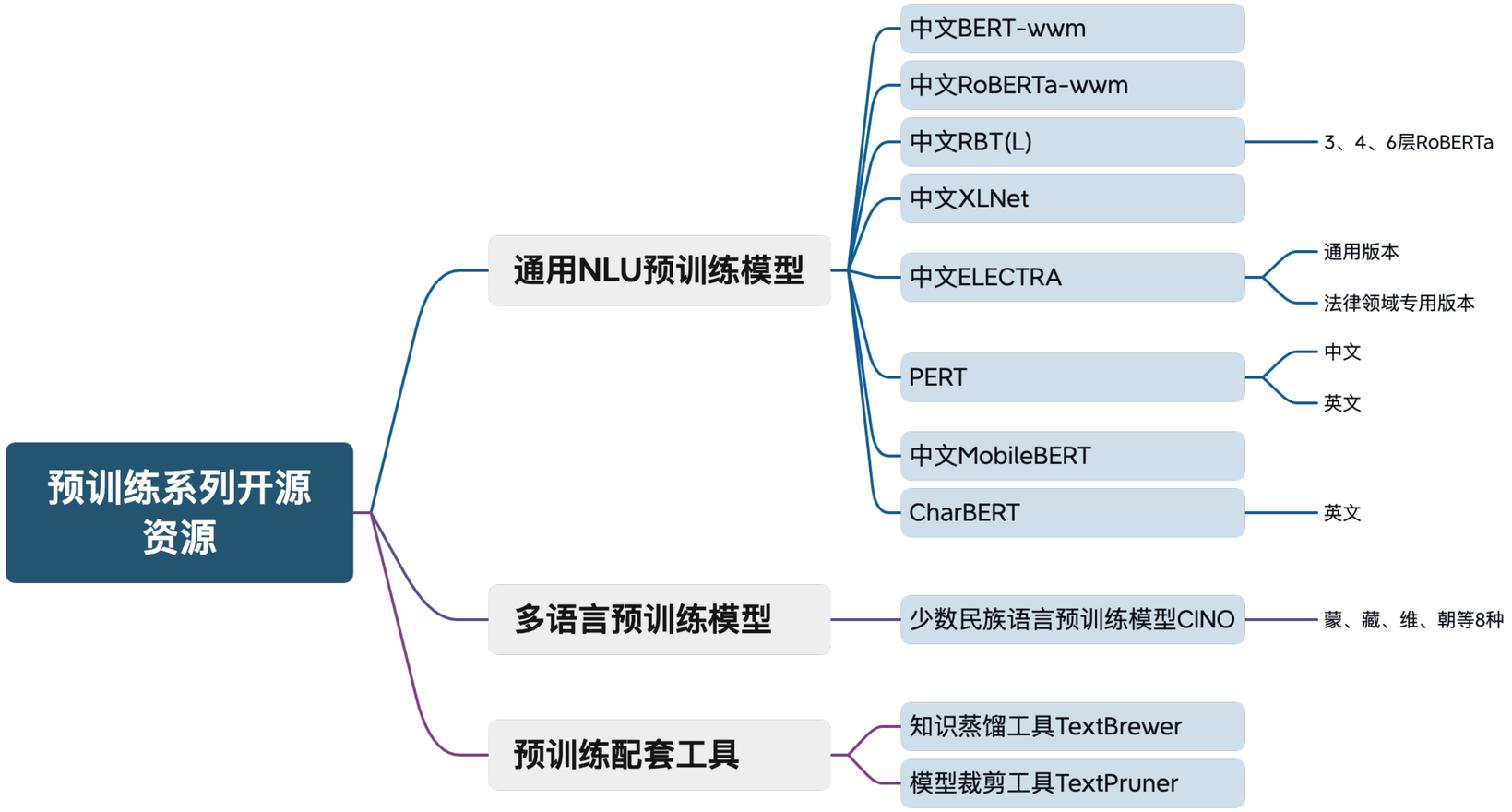
Model	CMRC 2018	XNLI
BERT-base	84.5 / 65.5	77.8
BERT-wwm (base)	85.6 / 66.3	79.0
BERT-wwm-ext (base)	85.7 / 67.1	79.4

▲ 中文任务效果

[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：中文BERT-wwm

- **构建预训练系列开源资源**
  - 依托相关技术，进一步完善并构建中文预训练系列模型并开源至研究社区



[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：MacBERT

- **基于纠错型MLM的预训练模型MacBERT**
  - 解决BERT中的“预训练-精调”不一致的问题
  - 提出了一种基于相似词替换的预训练任务（**M**LM **a**s **c**orrection，Mac），类似“文本纠错”

用 语 言 模 型 **预 测** 下 一 个 词

- 80% of the time, replace with [M]
  - 用 语 言 模 型 **[M] [M]** 下 一 个 词
- 10% of the time, replace random word
  - 用 语 言 模 型 **预 见** 下 一 个 词
- 10% of the time, keep the same word
  - 用 语 言 模 型 **预 测** 下 一 个 词

BERT

- 80% of the time, replace with synonym
  - 用 语 言 模 型 **预 见** 下 一 个 词
- 10% of the time, replace random word
  - 用 语 言 模 型 **好 是** 下 一 个 词
- 10% of the time, keep the same word
  - 用 语 言 模 型 **预 测** 下 一 个 词

MacBERT

[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：MacBERT

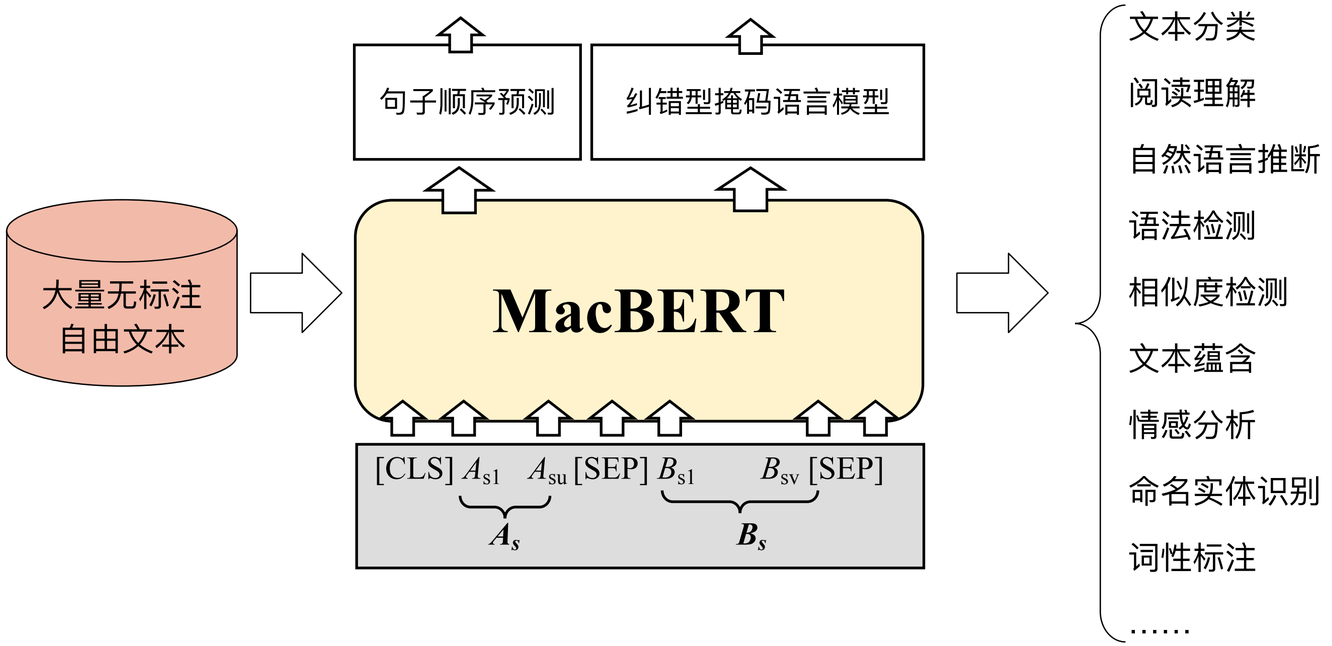
- **N元文法掩码（N-gram masking）**
  - 对一个连续的N-gram单元进行掩码，进一步增加MLM任务的难度
  - 预测难度：N-gram Masking > Whole Word Masking > vanilla MLM

原句	We went to the store to buy some fruits.
整词掩码	We went to the [M] to [M] some [M].
N-gram掩码	We went to the store to [M] [M] [M].

[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：MacBERT

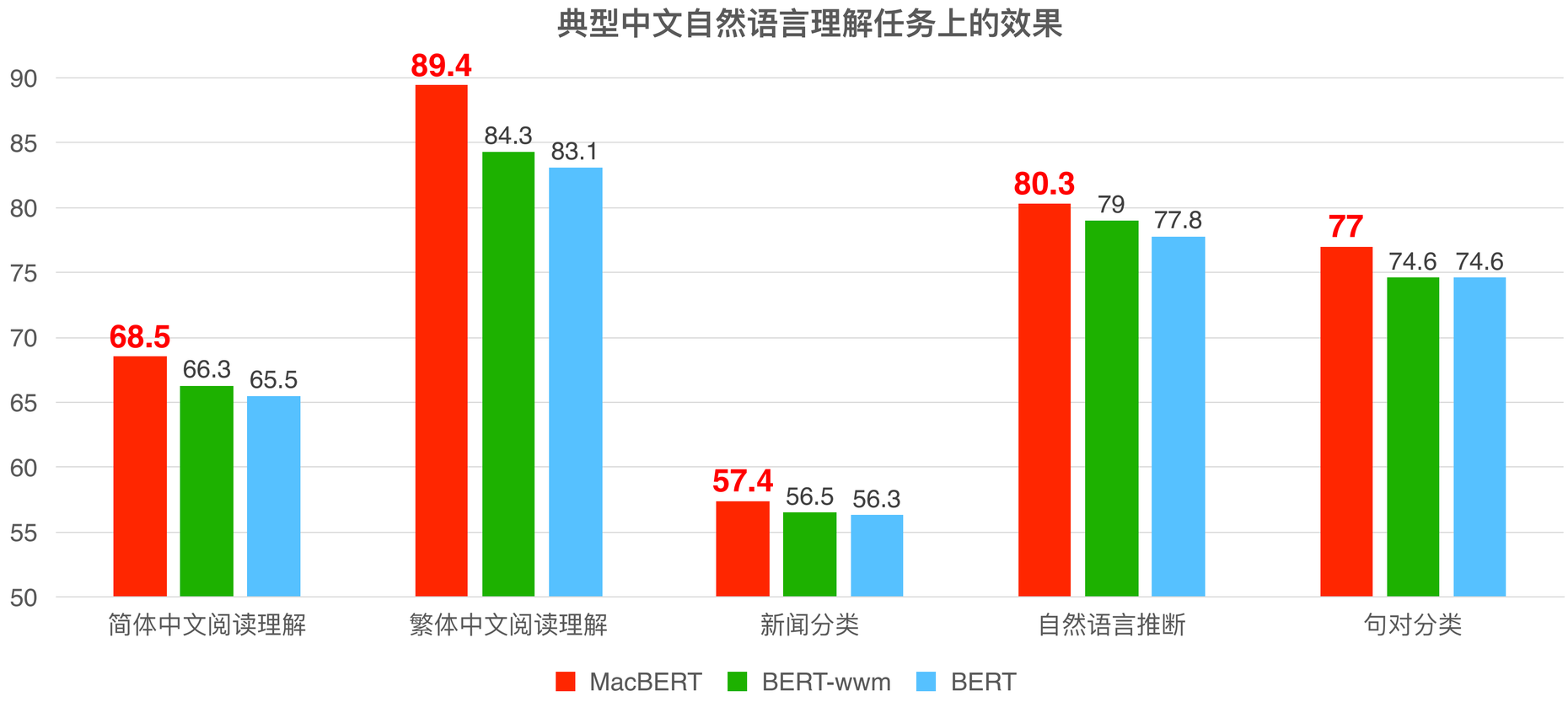
- **模型结构及训练任务**
  - **LM任务**：使用基于相似词替换的预训练任务进行训练
  - **SP任务**：将“下一个句子预测（NSP）”替换为ALBERT中的“句子顺序预测（SOP）”任务



[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：MacBERT

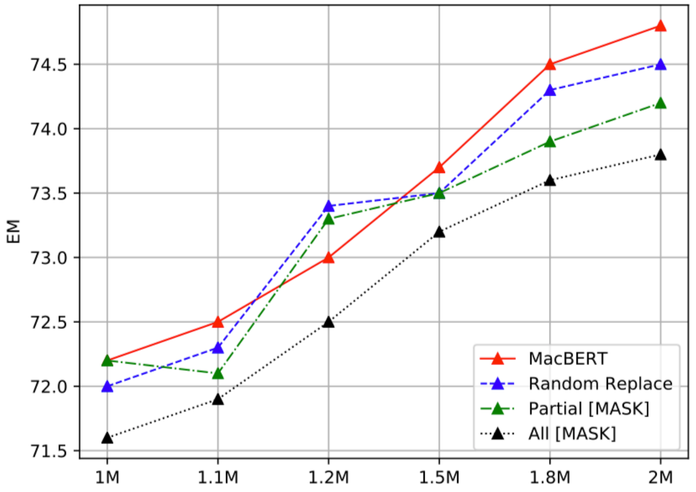
- **实验结果**
  - 在阅读理解、文本分类、自然语言推断等任务上获得显著性能提升



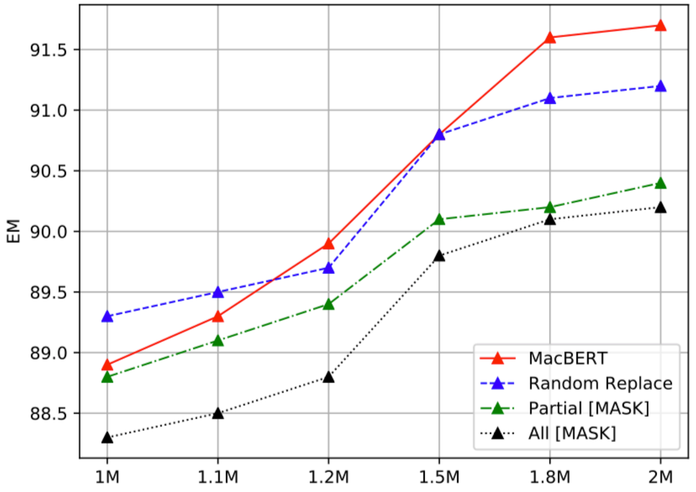
[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：MacBERT

- **实验分析：MLM任务**
  - 输入句子中15%的token被处理
    - MacBERT：80%的token被替换为相似词，10%被替换为随机词
    - Random Replace：90%的token被替换为随机词
    - Partial Mask：原始BERT实现，即80%的token被替换为 [MASK] 符号，10%被替换为随机词
    - All Mask：90%的token被替换为 [MASK]
  - 剩余的10%不做任何替换（负样本）
  - 效果排序
    - MacBERT > random replace > partial mask > all mask



CMRC 2018



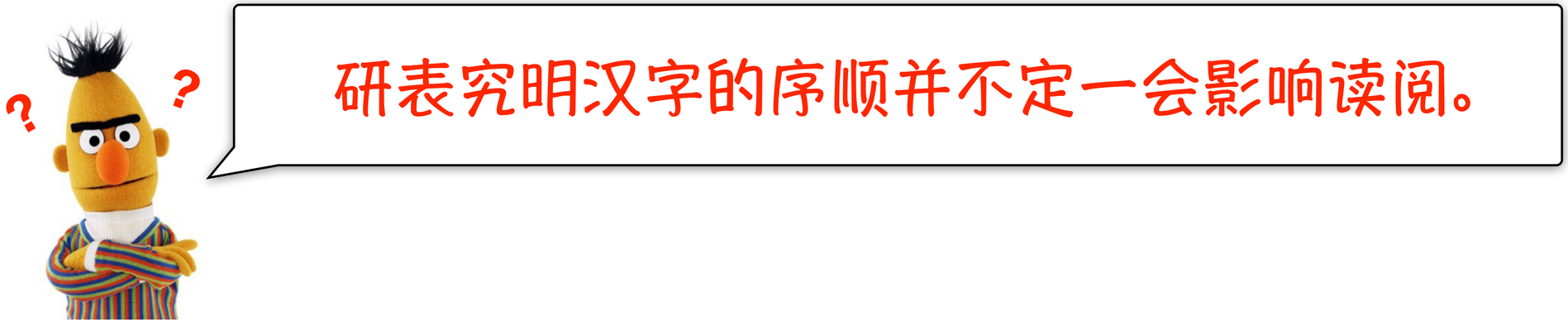
DRCD

▲ 横轴：训练步数；纵轴：EM值

[Cui et al., 2020] Revisiting Pre-trained Models for Chinese Natural Language Processing. In Findings of EMNLP 2020.
[Cui et al., 2021] Pre-training with Whole Word Masking for Chinese BERT. In IEEE/ACM TASLP Vol.28.

通用预训练语言模型：PERT

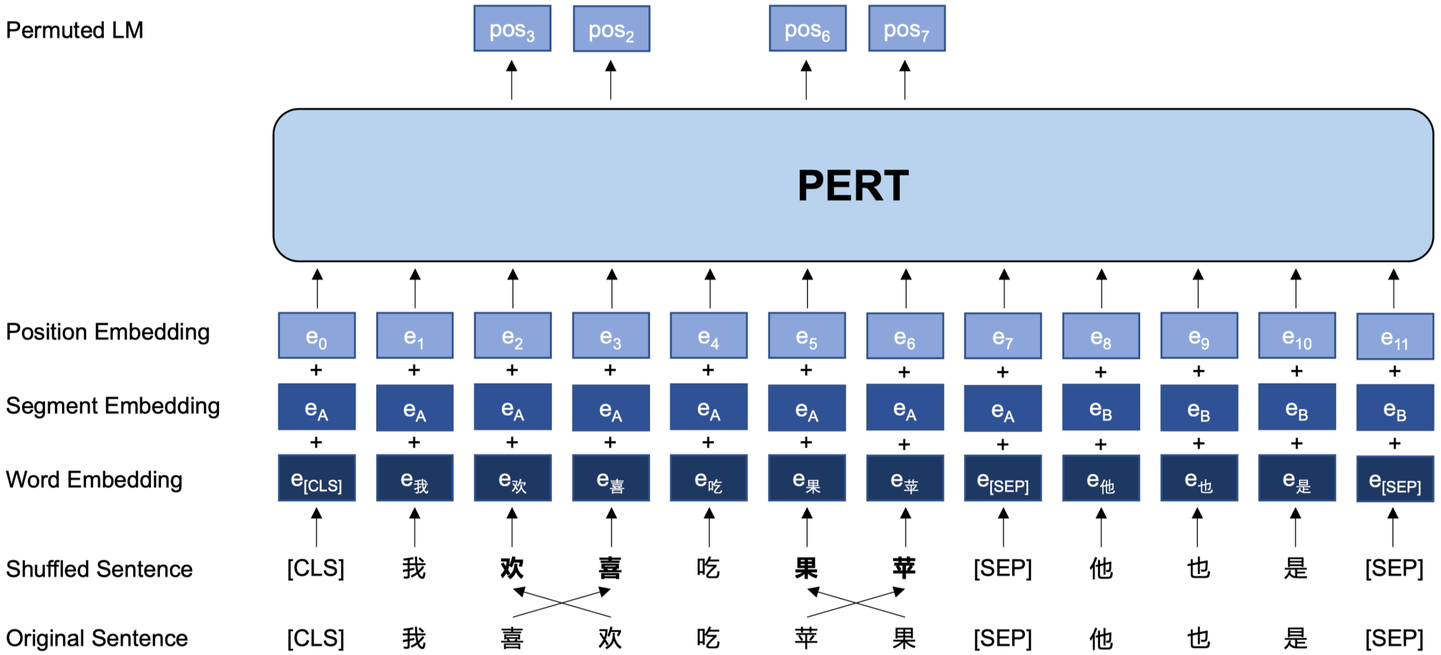
- **PERT: Pre-training BERT with Permuted Language Model**
  - 研究背景
    - 多数自编码预训练模型基于MLM及其变种进行训练
    - 基于MLM的训练方法依赖[MASK]标记，造成“预训练-精调”不一致的问题
  - 问题：有没有其他可以建模文本语义的方式?
  - 观察：发现乱序文本仍然包含与原文本相近的语义



[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

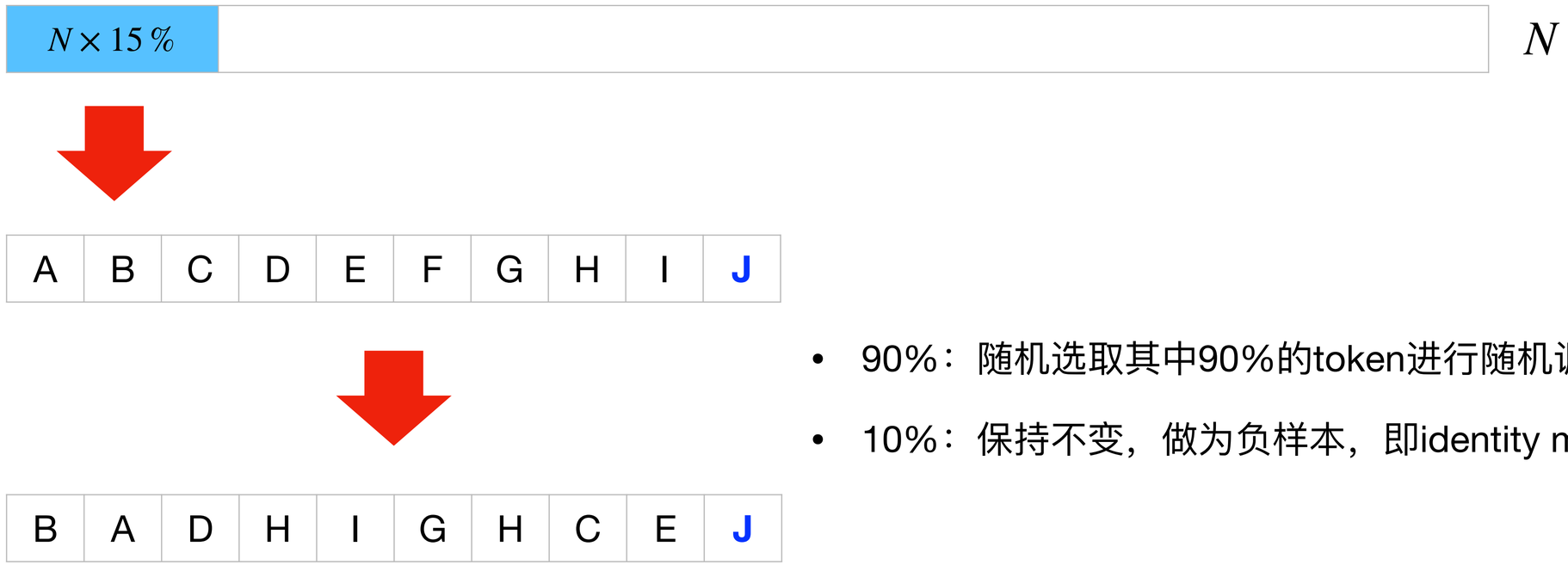
- **模型结构**
  - 提出利用乱序文本预测当前单词在原句的位置，自监督地学习文本语义信息
  - 该过程不会引入掩码标记[MASK]
  - 输出空间是输入序列长度N，而非词表大小V



[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **Permuted Language Model (PerLM)**
  - 随机选取输入序列长度15%的文本用于掩码
  - 同时使用全词掩码（wwm）和N-gram掩码技术（unigram 40% → 4-gram 10%）



- 90%：随机选取其中90%的token进行随机调序
- 10%：保持不变，做为负样本，即identity mapping

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **实验效果：中文NLU任务**
  - 使用20G文本训练了PERT模型，其中包含维基百科、新闻、社区问答等语料
  - 实验结果：PERT在机器阅读理解、命名实体识别任务上能够获得显著性能提升

System	CMRC 2018						DRCD			
	D-EM	D-F1	T-EM	T-F1	C-EM	C-F1	D-EM	D-F1	T-EM	T-F1
$BERT_{base}$	67.1 (65.6)	85.7 (85.0)	71.4 (70.0)	87.7 (87.0)	24.0 (20.0)	47.3 (44.6)	85.0 (84.5)	91.2 (90.9)	83.6 (83.0)	90.4 (89.9)
$RoBERTa_{base}$	67.4 (66.5)	87.2 (86.5)	72.6 (71.4)	89.4 (88.8)	26.2 (24.6)	51.0 (49.1)	86.6 (85.9)	92.5 (92.2)	85.6 (85.2)	92.0 (91.7)
$ELECTRA_{base}$	68.4 (68.0)	84.8 (84.6)	73.1 (**72.7**)	87.1 (86.9)	22.6 (21.7)	45.0 (43.8)	87.5 (87.0)	92.5 (92.3)	86.9 (86.6)	91.8 (91.7)
$MacBERT_{base}$	**68.5** (67.3)	**87.9 (87.1)**	**73.2** (72.4)	**89.5 (89.2)**	**30.2** (26.4)	54.0 (52.2)	89.4 (**89.2**)	**94.3 (94.1)**	**89.5 (88.7)**	**93.8 (93.5)**
$PERT_{base}$	**68.5 (68.1)**	87.2 (**87.1**)	72.8 (72.5)	89.2 (89.0)	28.7 (**28.2**)	**55.4 (53.7)**	**89.5** (88.9)	93.9 (93.6)	89.0 (88.5)	93.5 (93.2)
$RoBERTa_{large}$	68.5 (67.6)	88.4 (87.9)	74.2 (72.4)	90.6 (90.0)	31.5 (30.1)	60.1 (57.5)	89.6 (89.1)	94.8 (94.4)	89.6 (88.9)	94.5 (94.1)
$ELECTRA_{large}$	69.1 (68.2)	85.2 (84.5)	73.9 (72.8)	87.1 (86.6)	23.0 (21.6)	44.2 (43.2)	88.8 (88.7)	93.3 (93.2)	88.8 (88.2)	93.6 (93.2)
$MacBERT_{large}$	70.7 (68.6)	88.9 (88.2)	74.8 (73.2)	**90.7** (90.1)	31.9 (29.6)	**60.2** (57.6)	**91.2 (90.8)**	**95.6 (95.3)**	**91.7 (90.9)**	**95.6 (95.3)**
$PERT_{large}$	**72.2 (71.0)**	**89.4 (88.8)**	**76.8 (75.5)**	**90.7 (90.4)**	**32.3 (30.9)**	59.2 (**58.1**)	90.9 (**90.8**)	95.5 (95.2)	91.1 (90.7)	95.2 (95.1)

▲ 阅读理解效果

System	MSRA-NER (Test)			People's Daily (Dev)		
	P	R	F	P	R	F
$BERT_{base}$	95.2 (94.8)	95.4 (95.1)	95.3 (94.9)	95.3 (**95.1**)	95.3 (95.1)	**95.3** (**95.1**)
$RoBERTa_{base}$	95.3 (94.9)	95.6 (95.4)	95.5 (95.1)	94.9 (94.8)	95.3 (95.1)	95.1 (94.9)
$ELECTRA_{base}$	95.0 (94.5)	**95.9** (95.4)	95.4 (95.0)	94.8 (94.7)	95.3 (**95.2**)	95.1 (94.9)
$MacBERT_{base}$	95.2 (94.9)	95.4 (95.4)	95.3 (95.1)	94.9 (94.6)	95.6 (95.1)	95.2 (94.9)
$PERT_{base}$	**95.4** (**95.2**)	95.5 (**95.5**)	**95.6** (**95.3**)	**95.4** (**95.1**)	95.2 (95.0)	**95.3** (**95.1**)
$RoBERTa_{large}$	95.4 (95.3)	95.7 (95.7)	95.5 (95.5)	95.7 (95.4)	95.7 (95.4)	95.7 (95.4)
$ELECTRA_{large}$	94.9 (94.8)	95.5 (95.0)	95.0 (94.8)	94.8 (94.6)	95.3 (95.3)	94.9 (94.8)
$MacBERT_{large}$	96.3 (95.8)	96.3 (95.9)	**96.2** (95.9)	95.8 (95.6)	95.8 (**95.7**)	95.8 (95.7)
$PERT_{large}$	**96.4** (**95.9**)	**96.4** (**96.1**)	**96.2** (**96.0**)	**96.3** (**96.0**)	96.0 (**95.7**)	**96.1** (**95.8**)

▲ 命名实体识别效果

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **实验效果：中文语法纠错任务**
  - 针对语法中的文本乱序问题，利用PERT搭建序列标注模型进行实验
  - 在四个领域（维基、公文、海关、政法）下，PERT均显著优于所有基线系统效果

我每天一个吃苹果 ⟶ 我每天吃一个苹果

System	Wikipedia			Formal Doc.			Customs			Legal		
	P	R	F	P	R	F	P	R	F	P	R	F
$\text{BERT}^{\text{Google}}_{\text{base}}$	83.6	76.3	79.8	92.1	87.1	89.6	85.7	85.1	85.4	94.3	89.8	92.0
$\text{RoBERTa}_{\text{base}}$	84.2	76.9	80.4	92.6	87.7	90.1	86.8	85.9	86.3	94.6	90.0	92.2
$\text{ELECTRA}_{\text{base}}$	69.9	57.8	63.6	88.1	81.6	84.7	69.6	71.0	70.3	91.7	85.4	88.4
$\text{MacBERT}_{\text{base}}$	84.3	77.1	80.5	92.7	87.8	90.2	86.4	86.5	86.4	94.6	90.1	92.3
$\text{PERT}_{\text{base}}$	**86.5**	**79.5**	**82.9**	**93.6**	**89.0**	**91.2**	**88.3**	**88.0**	**88.2**	**95.2**	**90.7**	**92.9**

▲ 文本纠错-乱序问题效果

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **实验效果：英文NLU任务**
  - 使用了经典的Wikipedia+Books数据进行训练
  - 实验结果：部分任务上优于其他同等数据训练的预训练模型

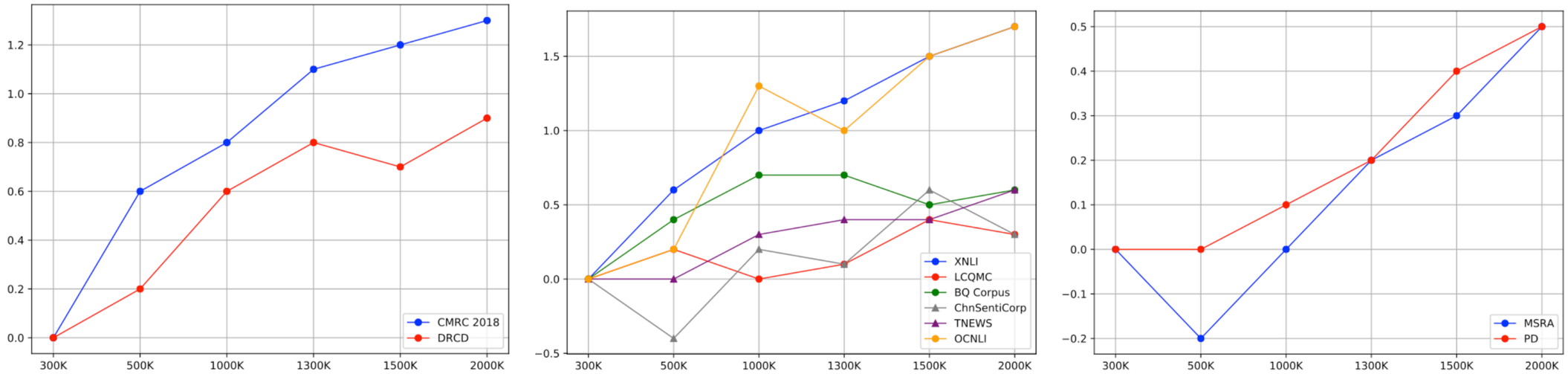
System	SQuAD EM	SQuAD F1	SQuAD 2.0 EM	SQuAD 2.0 F1	MNLI Acc	SST-2 Acc	CoLA M.C.	MRPC Acc
$\text{BERT}_{\text{base}}$	80.8	88.5	-	-	84.4	**92.7**	60.6	86.7
$\text{BERT}_{\text{base}}^{\dagger}$	81.2	88.5	72.4	75.4	84.4	92.6	59.3	86.0
$\text{RoBERTa}_{\text{base}}$	-	90.4	-	79.1	84.7	92.5	-	-
$\text{XLNet}_{\text{base}}$	-	-	**78.4**	**81.3**	**85.8**	92.6	-	-
$\text{ALBERT}_{\text{base}}$	82.1	89.3	76.1	79.1	81.9	89.4	-	-
$\text{PERT}_{\text{base}}$	**84.8**	**91.3**	78.3	81.0	84.5	92.0	**61.2**	**87.5**
$\text{BERT}_{\text{large}}$	84.1	90.9	78.7	81.9	86.6	93.2	60.6	88.0
$\text{BERT}_{\text{large-wwm}}^{\dagger}$	87.4	93.4	82.8	85.6	87.3	93.4	63.1	87.2
$\text{RoBERTa}_{\text{large}}$	-	93.6	-	87.3	**89.0**	**95.3**	-	-
$\text{XLNet}_{\text{large}}$	**88.2**	**94.0**	**85.1**	**87.8**	88.4	94.4	65.2	**90.0**
$\text{ALBERT}_{\text{large}}$	84.1	90.9	79.0	82.1	83.8	90.6	-	-
$\text{PERT}_{\text{large}}$	87.4	93.3	83.5	86.3	87.6	93.4	**65.7**	87.3

▲ 英文任务效果

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **不同训练步数下的性能对比**
  - 阅读理解、命名实体识别任务性能趋势基本一致，随着训练步数增加，其性能也呈现增长趋势
  - 多数文本分类任务在训练中途达到局部最优效果
  - “预训练”也和“下游任务精调”一样，最终时刻并不一定是最好结果



(a) MRC　　(b) TC　　(c) NER

▲ **横轴**：训练步数；**纵轴**：以300K效果为基准计算性能差值

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- **分析：不同调序粒度**
  - 在PERT中，被选中的词可以和任何一个词进行调换，对文本自然度破坏更大
  - 分析不同调序粒度对性能的影响：word，N-gram，sentence
  - 实验结果显示
    - 调序粒度越小，虽然对文本的自然度破坏更小，却不利于文本的语义学习
    - 对阅读理解任务影响较大，对文本分类任务影响较小

System	CMRC 2018						XNLI		TNEWS	OCNLI	Average
	D-EM	D-F1	T-EM	T-F1	C-EM	C-F1	Dev	Test	Dev	Dev	
$\text{PERT}_{\text{base}}$ (no limit)	65.4	85.0	70.2	87.3	22.4	45.6	74.8	74.4	54.5	70.6	65.02
└Word	59.3	80.6	64.8	83.5	12.6	32.2	73.2	72.1	53.2	69.3	60.08
└N-gram	62.2	82.5	67.3	84.8	17.2	36.1	73.4	73.2	53.8	69.5	62.00
└Sentence	63.7	83.2	69.1	86.2	16.8	38.0	74.3	73.0	54.1	70.0	62.84

▲ 不同调换粒度下的实验结果

[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

通用预训练语言模型：PERT

- 分析：**Global v.s. Local Prediction**
  - 在PERT框架下，在局部空间预测效果相对较好，结合两者并未带来额外的性能提升

System	CMRC 2018 D-EM	CMRC 2018 D-F1	CMRC 2018 T-EM	CMRC 2018 T-F1	CMRC 2018 C-EM	CMRC 2018 C-F1	XNLI Dev	XNLI Test	TNEWS Dev	OCNLI Dev	Average
PERT$_{base}$ (local)	64.1	84.0	69.1	86.5	21.0	43.3	74.1	74.4	54.5	70.6	64.16
└Global	61.1	81.4	65.8	84.3	15.8	36.2	73.6	74.0	55.4	69.4	61.70
└Local + Global	63.2	83.6	67.7	85.8	19.3	42.0	74.6	74.6	55.1	70.0	63.59

▲ **Global**：在整个词表上预测，即 $y \in \mathbb{R}^{|V|}$，**Local**：在输入句子中预测，即 $y \in \mathbb{R}^{L}$

- 分析：**Partial v.s. Full Prediction**
  - Full Prediction未能在PERT框架下带来更好的实验结果
  - 并非所有预训练任务都适合ELECTRA-style预测方式，即Full Prediction

System	CMRC 2018 D-EM	CMRC 2018 D-F1	CMRC 2018 T-EM	CMRC 2018 T-F1	CMRC 2018 C-EM	CMRC 2018 C-F1	XNLI Dev	XNLI Test	TNEWS Dev	OCNLI Dev	Average
PERT$_{base}$ (partial)	64.1	84.0	69.1	86.5	21.0	43.3	74.1	74.4	54.5	70.6	64.16
└Full Prediction	63.7	84.1	68.0	86.1	18.7	40.9	74.3	73.9	54.2	71.0	63.49

▲ **Partial**：只对调换的token进行预测，**Full**：对输入序列中的所有token进行预测

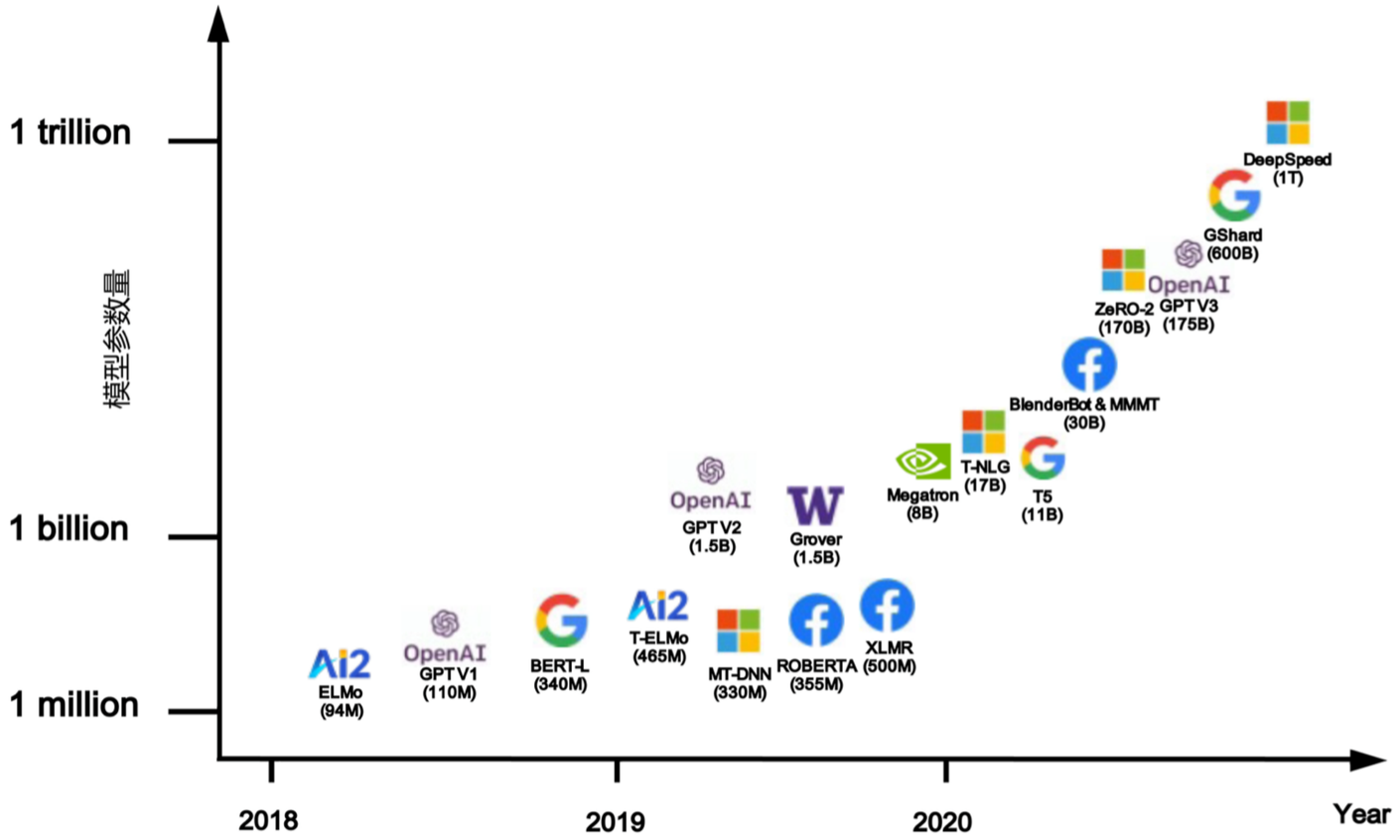
[Cui et al., 2022] PERT: Pre-training BERT with Permuted Language Model. arXiv pre-print.

高效模型训练与推理

Efficient Model Training and Inference

高效模型训练与推理

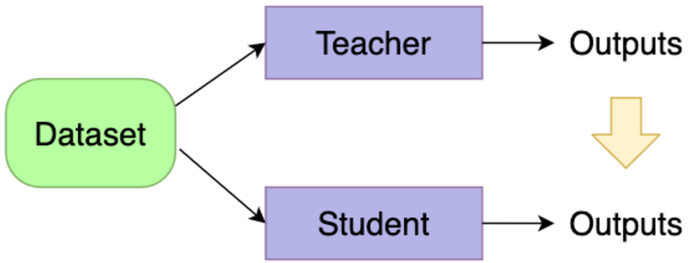
- **研究背景**
  - 预训练模型通常需要占用很大的空间，并且训练和推断时间也很慢，难以满足实际应用需求



高效模型训练与推理

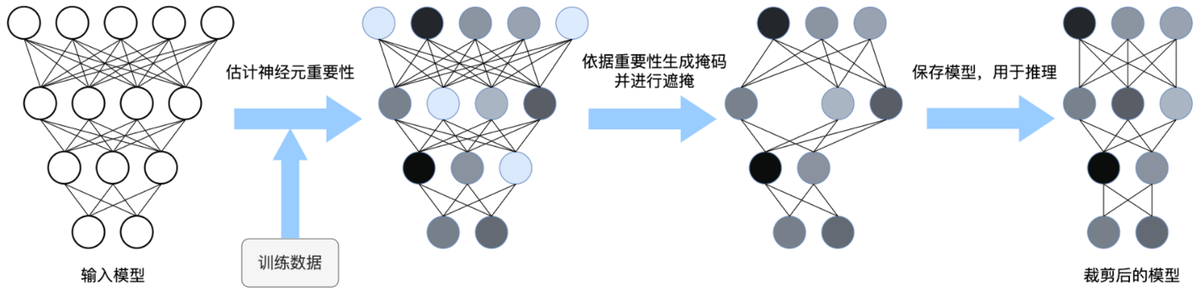
- **高效模型训练与推理的主要途径**
  - **知识蒸馏**：在少量性能损失的情况下，将大模型知识迁移到小模型，提升模型效果和推断速度
  - **模型裁剪**：对模型中的部分“不重要”或“冗余”的部分进行剪枝，从而缩小预训练模型体积

知识蒸馏



TextBrewer

模型裁剪



TextPruner

高效模型训练与推理

- **TextBrewer: An Open-Source Knowledge Distillation Toolkit for NLP**
  - 推出了首个面向自然语言处理领域的基于PyTorch的知识蒸馏工具包
  - 提供了方便、快捷、易用的知识蒸馏框架，少量性能损失换取大幅速度提升
  - 访问 http://textbrewer.hfl-rc.com/ 或通过 `pip install textbrewer`安装

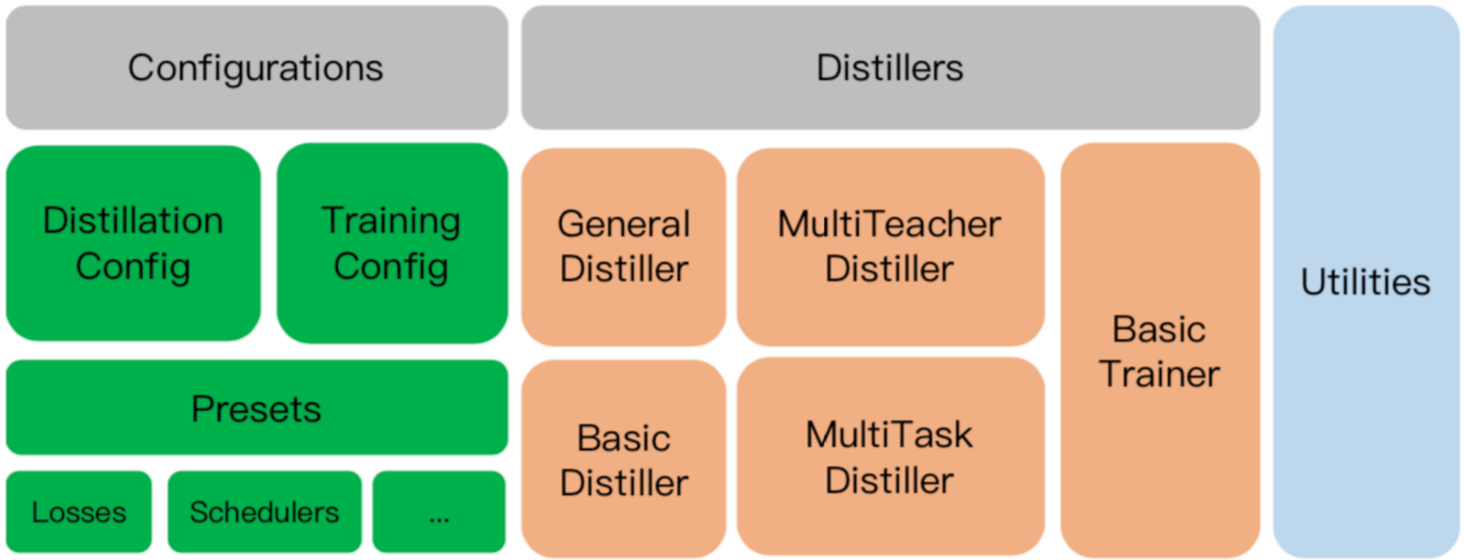
TextBrewer 1110+ ⭐

- **模型无关**：适用于多种模型结构（主要面向Transfomer结构）
- **方便灵活**：可自由组合多种蒸馏方法，支持增加自定义损失等模块
- **非侵入式**：无需对教师与学生模型本身结构进行修改
- **适用面广**：支持典型NLP任务，如文本分类、阅读理解、序列标注等

[Yang et al., 2020] TextBrewer: An Open-Source Knowledge Distillation Toolkit for Nature Language Processing. In ACL 2020.

高效模型训练与推理

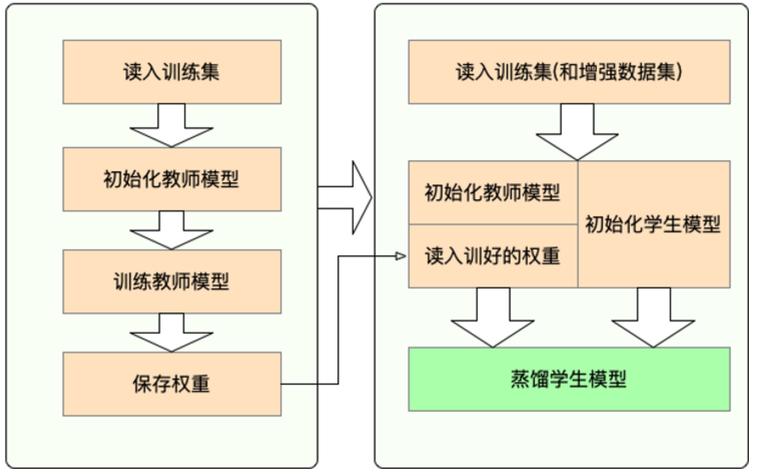
- **工具包设计架构**
  - Distillers：用于执行实际的知识蒸馏工作，定义了常规蒸馏策略
  - Configurations：为Distillers提供必要的配置信息
  - Utilities：包含一些辅助的功能，如模型参数统计等



[Yang et al., 2020] TextBrewer: An Open-Source Knowledge Distillation Toolkit for Nature Language Processing. In ACL 2020.

高效模型训练与推理

- 工作流程
  - **第一步：蒸馏之前的准备工作**
    - 训练教师模型，定义并初始化学生模型
    - 构造蒸馏用数据集的DataLoader
  - **第二步：知识蒸馏**
    - 初始化Distiller，构造训练配置和蒸馏配置
    - 定义adaptors和callback，分别用于适配模型输入输出和训练过程中的回调
    - 调用Distiller的train方法开始蒸馏
  - 蒸馏已经预设的相关任务，只需要20行左右的代码！




```python
from textbrewer import GeneralDistiller
from textbrewer import TrainingConfig, DistillationConfig

# We omit the initialization of models, optimizer, and dataloader.
teacher_model : torch.nn.Module = ...
student_model : torch.nn.Module = ...
dataloader : torch.utils.data.DataLoader = ...
optimizer : torch.optim.Optimizer = ...
scheduler : torch.optim.lr_scheduler = ...

def simple_adaptor(batch, model_outputs):
    # We assume that the first element of model_outputs
    # is the logits before softmax
    return {'logits': model_outputs[0]}

train_config = TrainingConfig()
distill_config = DistillationConfig()
distiller = GeneralDistiller(
    train_config=train_config, distill_config = distill_config,
    model_T = teacher_model, model_S = student_model,
    adaptor_T = simple_adaptor, adaptor_S = simple_adaptor)

distiller.train(optimizer, scheduler,
    dataloader, num_epochs, callback=None)
```


[Yang et al., 2020] TextBrewer: An Open-Source Knowledge Distillation Toolkit for Nature Language Processing. In ACL 2020.

高效模型训练与推理

- **知识蒸馏效果**
  - 教师模型：BERT-base (110M)
  - 学生模型
    - T6 (60%), T3 (41%), T3-small (16%), T4-tiny (same as TinyBERT, 13%)
  - 单教师知识蒸馏
    - T6结构可以达到教师模型效果的99%，模型体积缩小至60%
    - T4-tiny知识蒸馏结果优于TinyBERT
  - 多教师知识蒸馏
    - 蒸馏后的学生模型获得最优效果，且超过简单的模型融合方法（ensemble）

Model	MNLI		SQuAD		CoNLL-2003
	m	mm	EM	F1	F1
$\text{BERT}_{\text{BASE}}$	83.7	84.0	81.5	88.6	91.1
Public					
DistilBERT	81.6	81.1	79.1	86.9	-
TinyBERT	80.5	81.0	-	-	-
+DA	82.8	82.9	72.7	82.1	-
TextBrewer					
BiGRU	-	-	-	-	85.3
T6	83.6	84.0	80.8	88.1	90.7
T3	81.6	82.5	76.3	84.8	87.5
T3-small	81.3	81.7	72.3	81.4	78.6
T4-tiny	82.0	82.6	73.7	82.5	77.5

▲ 单教师蒸馏效果

Model	MNLI		SQuAD		CoNLL-2003
	m	mm	EM	F1	F1
Teacher 1	83.6	84.0	81.1	88.6	91.2
Teacher 2	83.6	84.2	81.2	88.5	90.8
Teacher 3	83.7	83.8	81.2	88.7	91.3
Ensemble	84.3	84.7	82.3	89.4	91.5
Student	**84.8**	**85.3**	**83.5**	**90.0**	**91.6**

▲ 多教师蒸馏效果

[Yang et al., 2020] TextBrewer: An Open-Source Knowledge Distillation Toolkit for Nature Language Processing. In ACL 2020.

高效模型训练与推理

- **TextPruner: A Model Pruning Toolkit for Pre-trained Language Model**
  - 推出首个面向自然语言处理领域的基于PyTorch的模型裁剪工具包
  - 通过轻量、快速的裁剪方法对模型进行结构化剪枝，从而实现压缩模型体积、提升模型速度
  - 访问 http://textpruner.hfl-rc.com/ 或通过 `pip install textpruner`安装

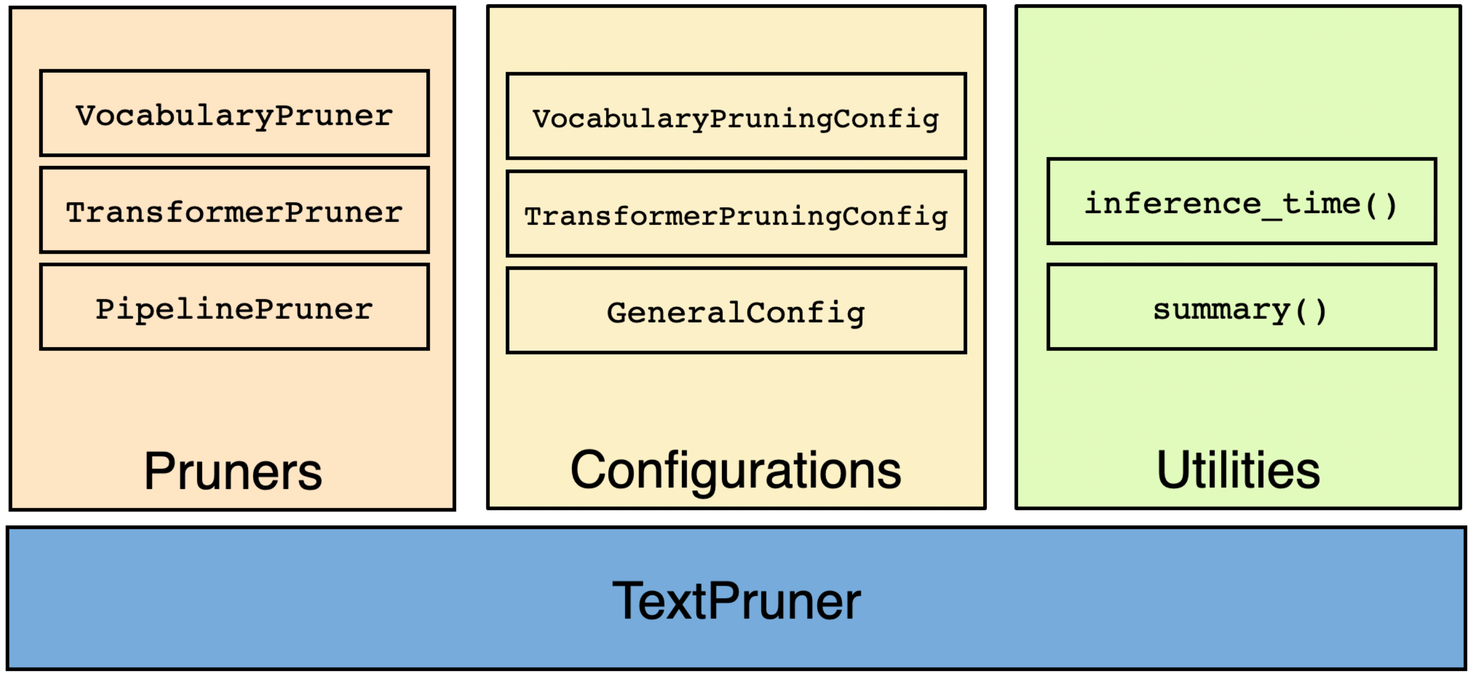
TextPruner 180+ ⭐

- **功能通用：**适配多种预训练模型，以及多种NLU任务
- **可定制化：**除了标准PLM外，也可使用TextPruner裁剪基于标准PLM开发的自定义模型
- **灵活便捷：**可作为Python包在Python脚本中使用，也提供了独立的命令行工具
- **运行高效：**使用无训练的结构化裁剪方法，运行迅速，快于知识蒸馏等基于训练的方法

[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

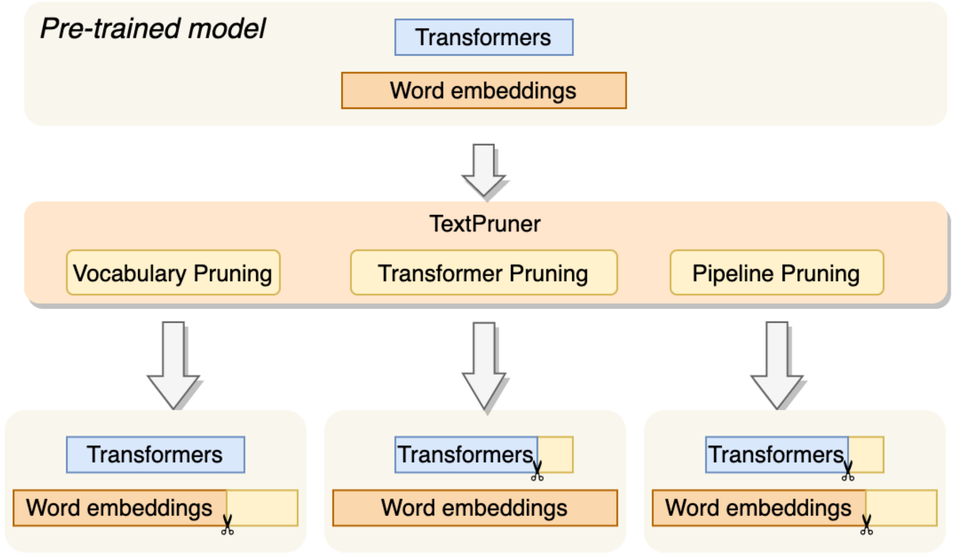
- **工具包设计架构**
  - Pruners：用于执行实际的模型裁剪，包含词表裁剪、Transformer裁剪、流水线裁剪等
  - Configurations：为Distillers提供必要的配置信息
  - Utilities：包含一些辅助的功能，如模型参数统计、计算推断时间等



[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

- **裁剪模式**
  - **词表裁剪**：移除词表中未被使用的单词，实现减小模型体积，提升MLM任务速度的效果
  - **Transformer裁剪**：裁剪“不重要”的注意力头和全连接层神经元，在减小模型体积的同时把对模型性能的影响尽可能降到最低
  - **流水线裁剪**：依次分别进行Transformer裁剪和词表裁剪，对模型体积做全面的压缩



☞ 计算“重要性” $\mathrm{IS}(\mathbf{\Theta}) = \mathbb{E}_{x\sim X}\left|\frac{\partial \mathcal{L}(x)}{\partial \mathbf{\Theta}}\mathbf{\Theta}\right|$

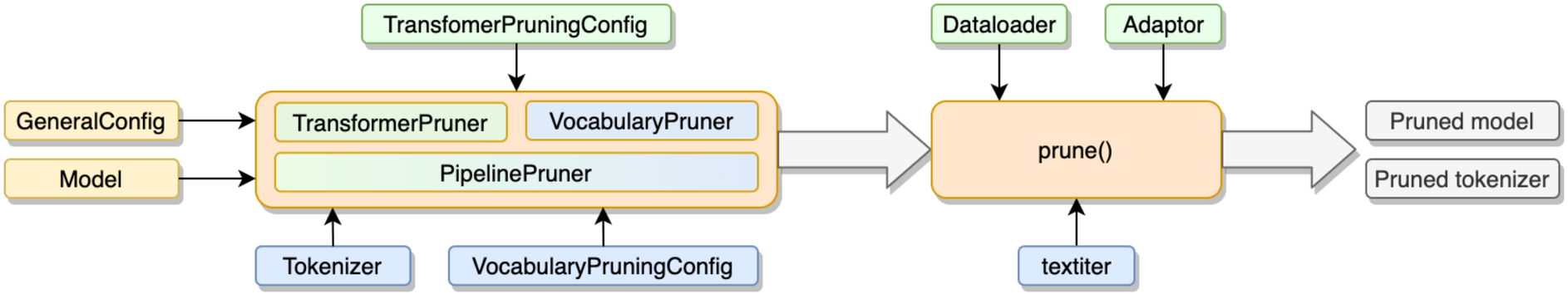
☞ 有监督裁剪 $\mathscr{L}_{CE}(x) = -\sum_{x} y \log q(x)$

☞ 无监督裁剪 $\mathscr{L}_{KL}(x) = \mathbf{KL}(\mathbf{stopgrad}(q(x))||p(x))$

[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

- 裁剪方法



Pruning with Python API

```
from textpruner import PipelinePruner
from textpruner import TransformerPruningConfig

config = TransformerPruningConfig(
    pruning_method='iterative', n_iters=4,
    target_ffn_size=2048,target_num_of_heads=8)
pruner = PipelinePruner(model, tokenizer,config)
pruner.prune(dataloader=dataloader, dataiter=texts)
```

▲ 通过Python API裁剪

Pruning with CLI

```
textpruner-cli \
 --pruning_mode pipeline \
 --configurations vc.json trm.json \
 --model_class BertForClassification \
 --tokenizer_class BertTokenizer \
 --model_path models \
 --vocabulary texts.txt \
 --dataloader_and_adaptor dl.py
```

▲ 通过命令行裁剪

[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

- **模型裁剪效果：词表裁剪**
  - 实验设置
    - 实验以多语言预训练模型XLM-R在自然语言推断任务XNLI为基准进行测试
    - 使用英文数据训练，使用中文数据测试 (zero-shot)
    - 词表裁剪后，预训练模型只保留对应语种的单词
  - 实验结果表明，缩减词表后模型体积减小60%左右，基本保持了与基线系统持平的效果
  - 在指定语种的情况下，词表裁剪特别适合应用在多语言预训练模型

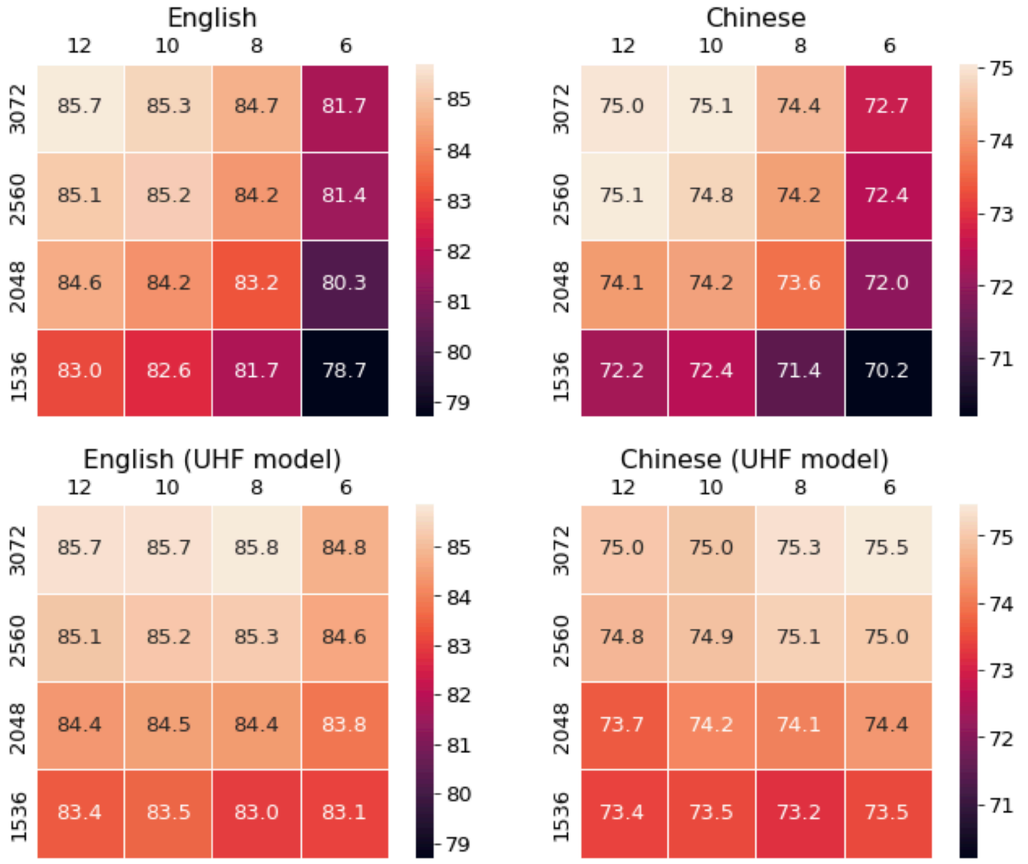
	Model	Vocabulary size	Model size	Dev (en)	Dev (zh)	Test (en)	Test (zh)
基模型	XLM-R	250002	1060 MB (100%)	84.8	75.1	85.7	75.0
裁剪模型	+ Vocabulary Pruning on en	26653	406 MB (38.3%)	84.6	-	85.9	-
	+ Vocabulary Pruning on zh	23553	397 MB (37.5%)	-	74.7	-	74.5
	+ Vocabulary Pruning on en and zh	37503	438 MB (41.3%)	84.8	74.3	85.8	74.5

▲ 词表裁剪效果

[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

- **模型裁剪效果：Transformer裁剪**
  - 使用XNLI英文开发集计算Importance Scores
  - 两种裁剪方法
    - UHF (**U**neven attention **H**eads and **F**FN sizes)
    - HP (**H**omogenous **P**runing)
  - 观察结论
    - UHF相比HP具有更大的灵活度，能够获得更好的裁剪效果
    - 中文测试集（zero-shot）上的性能损失与英文相近
      - 对中文任务重要的神经元仍然被保留
      - 提供通用跨语言理解能力的神经元被保留

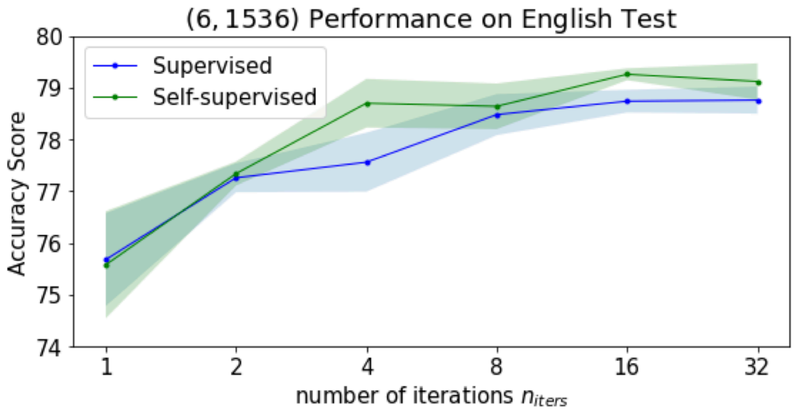


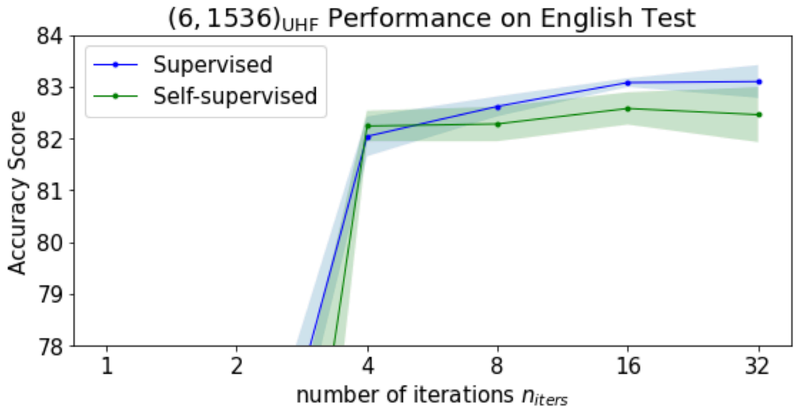
▲ XNLI测试集结果，横轴：平均注意力头数，
纵轴：FFN大小

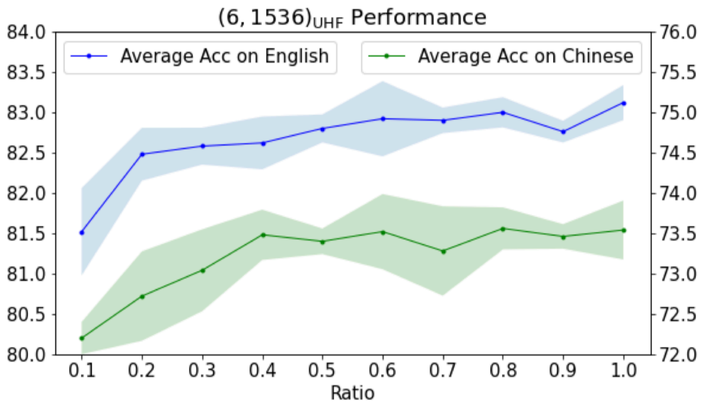
[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

高效模型训练与推理

- **分析：有监督 v.s. 无监督（Fig. 1&2）**
  - 自监督方法能够在不利用标签的情况下达到与有监督方法可比甚至更优的实验结果
  - 为了保证IS计算准确，建议迭代轮数不低于8轮
- **分析：需要多少数据来计算IS？（Fig. 3）**
  - 当使用70%的开发集用于计算IS，其效果与使用全量开发集达到可比状态







[Yang et al., 2022] TextPruner: A Model Pruning Toolkit for Pre-trained Language Models. In ACL 2022.

总结

SUMMARY

总结

- **本期报告**
  - 通用预训练语言模型
    - 中文BERT-wwm及其系列模型
    - 利用纠错型掩码语言模型训练的MacBERT
    - 基于乱序文本的预训练模型PERT
  - 模型蒸馏与加速
    - 推出了两个工具包：TextBrewer、TextPruner
    - 提供了方便易用的接口，快速高效地完成知识蒸馏和模型裁剪
- **更多内容：多语言、多模态、可解释**
  - 欢迎参加SMP 2022讲习班（8月19日，在线举行）

相关资源

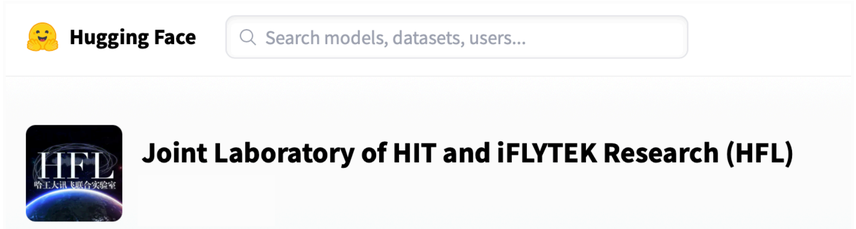
GitHub

- 中英文预训练模型：BERT-wwm, XLNet, ELECTRA, MobileBERT, MacBERT, PERT, CINO等
- 工具包：TextBrewer, TextPruner等
- 数据集：阅读理解、文本分类、文本纠错等
- HFL典藏集：https://github.com/ymcui/HFL-Anthology

哈工大讯飞联合实验室
微信公众号

Model Hub

- 搭配transformers库，快速加载各类预训练模型
- 目前已开放44个预训练语言模型
- 地址：https://huggingface.co/hfl




```
from transformers import BertTokenizer, BertModel

tokenizer = BertTokenizer.from_pretrained("hfl/
chinese-bert-wwm")
model = BertModel.from_pretrained("hfl/chinese-bert-
wwm")
```

THANK YOU!

https://github.com/ymcui

https://ymcui.com

me@ymcui.com

科大讯飞 iFLYTEK