计算机科学
COMPUTER SCIENCE

# 面向国产深度学习平台的自然语言处理模型迁移研究

葛慧斌, 王德鑫, 郑涛, 张婷, 熊德意

引用本文

葛慧斌, 王德鑫, 郑涛, 张婷, 熊德意. 面向国产深度学习平台的自然语言处理模型迁移研究[J]. 计算机科学, 2024, 51(1): 50-59.

GE Huibin, WANG Dexin, ZHENG Tao, ZHANG Ting, XIONG Deyi. Study on Model Migration of Natural Language Processing for Domestic Deep Learning Platform [J]. Computer Science, 2024, 51(1): 50-59.

---

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230600051

# 面向国产深度学习平台的自然语言处理模型迁移研究

葛慧斌[1]　王德鑫[1]　郑　涛[2]　张　婷[3]　熊德意[1]
1 天津大学智能与计算学部　天津 300350
2 华为技术有限公司南京研究所　南京 210000
3 中译语通科技股份有限公司　北京 100131
(gehuibin@tju.edu.cn)

**摘　要**　深度学习平台在新一代人工智能的发展中扮演着重要的角色。近年来,以昇腾平台为代表的国产人工智能软硬件系统快速发展,为国产深度学习平台的发展开辟出了新的道路。与此同时,为了发现并解决昇腾系统存在的潜在漏洞,昇腾平台积极开展常用深度学习模型的迁移工作。从自然语言处理算法角度切入,针对机器阅读理解、神经机器翻译、序列标注和文本分类四大自然语言处理任务,以昇腾平台的高性能硬件芯片为基础,探究迁移 ALBERT,RNNSearch,BERT-CRF 和 TextING 这 4 类典型的自然语言处理模型。基于以上迁移研究,发现和整理了昇腾平台架构设计在自然语言处理研究与业务上的主要不足,即计算图节点动态空间的分配特性、资源算子下沉设备侧、图算融合以及混合精度训练 4 个方面的问题,并为以上问题提出了相应的解决方案,并进行了实验验证。最后,为国产深度学习平台的发展提出未来优化的方向和相关建议。

**关键词**:自然语言处理;昇腾;深度学习;模型迁移;平台构架
**中图分类号**　TP183

## Study on Model Migration of Natural Language Processing for Domestic Deep Learning Platform

GE Huibin[1],WANG Dexin[1],ZHENG Tao[2],ZHANG Ting[3] and XIONG Deyi[1]
1 College of Intelligence and Computing,Tianjin University,Tianjin 300350,China
2 Nanjing Research Institute,Huawei Technologies Co. Ltd.,Nanjing 210000,China
3 Global Tone Communication Technology Co.,Ltd.,Beijing 100131,China

**Abstract**　Deep learning platformplays an essential role in the development of the new generation of artificial intelligence. In recent years,the domestic artificial intelligence high-performance software and hardware system of China represented by the Ascend platform has developed rapidly,which opens up a new way for the deep learning platform in China. At the same time,in order to explore and solve the potential loopholes in the Ascend system,the platform developers of Ascend actively carries out the migration of commonly used deep learning models with researchers. These efforts are further promoted from the perspective of natural language processingaiming at how to refine the domestic deep learning platform. Four natural language processing tasks arehighlighted,neural machine translation,machine reading comprehension,sequence labeling and text classification,along with four classical neural models,Albert,RNNSearch,BERT-CRF and TextING. They are migrated on the Ascend platform in details. Based on the above model migration research,this paper integrates the deficiencies of the architecture design of the Ascend platform in the research and business in natural language processing. In conclusion,these deficiencies are sorted out as four essential aspects:1)the lack of the dynamic space allocation characteristics of computing graph nodes;2)incompatibility for the sinking of resource operators on the acceleration-deviceside;3)the fusion of graphics and computing which is not flexible to handle unseen model structures,and 4)the defects of the mixed-precision training strategy. To overcome these problems,this paper puts forward the avoidance methods or solutions. Finally,constructive suggestions are provided for,including but not limited to,the deep-learning platforms in China.

**Keywords**　Natural language processing,Ascend,Deep learning,Model migration,Platform architecture

到稿日期:2023-06-06　返修日期:2023-10-07
基金项目:华为技术有限公司与天津大学 NRE 合作项目(20101300441922C);国家重点研发计划(2020AAA0108000);云南省重点研发计划(202203AA080004)
This work was supported by the Huawei Technologies Co.,Ltd. and Tianjin University NRE Cooperation Project(20101300441922C),National Key R&D Program of China(2020AAA0108000)and Key R&D Program of Yunnan Province(202203AA080004).
通信作者:熊德意(dyxiong@tju.edu.cn)

## 1 引言

得益于数据规模的扩大以及计算机算力的提升，深度学习在人工智能领域发挥着越来越重要的作用。深度学习神经网络涉及大量的矩阵运算，相比善于并行计算的图形处理单元(GPU)芯片，CPU 的矩阵运算效率较为低下。因此，基于 GPU 服务器的计算架构在全球深度学习服务器生态中占据主要地位，例如 Google 的开源深度学习框架 Tensorflow[1)]、Facebook 的 Pytorch[2)]等大多是基于 GPU 计算架构设计开发的。虽然 GPU 比 CPU 处理器更能高效计算神经网络，但是其优化需兼顾图像处理环节，这使得 GPU 在神经网络计算的优化方面受到一定限制。为了解决这个问题，谷歌于 2016 年发布了专门为 Tensorflow 定制的张量处理单元(TPU)，该芯片结合 Tensorflow 框架进行协同优化，能够更高效地训练深度学习模型，极大缩短了模型训练的时间。TPU 芯片被用于 AlphaG0 和 AlphaZero[3)]系统的训练和推理。得益于 TPU 对深度学习模型的加速和优化，当前许多大规模预训练模型能够以较短的时间完成训练，如 Google 于 2018 年发布的基于 TPU 训练的 BERT 模型[1-2]。

为了摆脱对国外深度学习软硬件平台的依赖，近年来，国内多家企业积极研发国产自主的深度学习硬件，构建自主研发的国产深度学习生态。2016 年，寒武纪推出国际上首个商用深度学习处理器 Cambricon-1A，这是国际上首个商用深度学习处理器产品[3]。2018 年，华为采用了寒武纪 IP，正式推出基于自研达芬奇架构的昇腾 AI 处理器 NPU，拥有接近于四核 CPU 25 倍以上的性能和 50 倍以上的能效[4]；2019 年，华为继续发布高性能昇腾 910 芯片，其算力进一步提高，步入国际前列。昇腾 910 芯片专门为神经网络计算进行针对性优化[4)]，提高了神经网络中基础运算的计算效率，开辟了国产 AI 芯片的新生态[5]。同年，阿里巴巴发布其首款 AI 芯片含光 800[5)]。含光 800 基于阿里云自主研发的硬体架构，专门优化阿里巴巴生态系统对内涉及的演算法[6]。2019 年 12 月，百度宣布首款用于云计算和边缘计算的 AI 芯片昆仑 1 代完成研发，该芯片基于 XPU 架构，专门为深度学习算法的云端和边缘端的计算而设计；2021 年 8 月，百度开始量产昆仑 2 代芯片[6)]。

除了硬件层面的探索之外，在软件层面上，如何结合硬件形成国产化 AI 生态，同样是亟待解决的问题。2016 年，百度开源国内首个功能完备的基于 GPU 服务器生态的开源深度学习平台 PaddlePaddle[7]；2020 年，华为基于高性能昇腾 910 芯片，开源 MindSpore 深度学习框架。MindSpore 是首个功能完备的基于国产化 AI 生态环境并覆盖全场景的深度学习框架。如图 1 所示，在硬件基座层面，昇腾研制了多款统一的、可扩展的 AI 处理器芯片。在基础软件层面，华为针对 AI 场景推出异构计算架构 CANN(Computer Architecture for Neural Network)，提供多层次编程接口，通过开发 AscendCL 和 TBE 编程接口，使不同 AI 应用可在 CANN 平台上高效快速地运行。此外，新一代深度学习训练框架 MindSpore 支持端、边、云统一协调、自动并行、自动微分及自动调优，以提高深度学习模型的训练效率，同时也有助于 AI 从业者进行编码调试[8]。

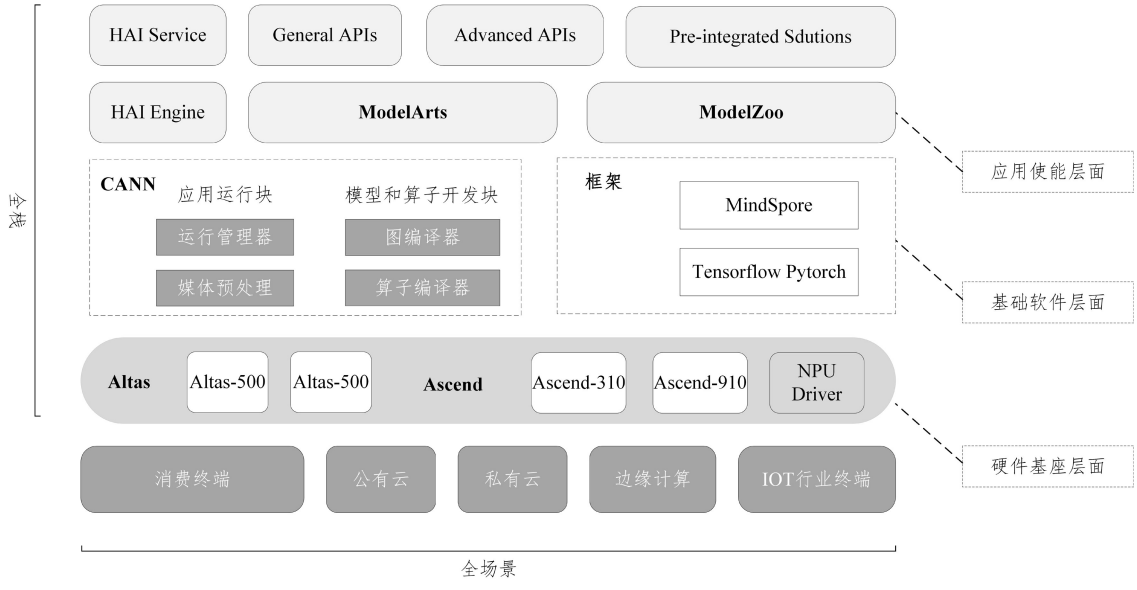


图 1 昇腾系统架构图[7)]

Fig. 1 Architecture of Ascend platform[7)]

大多数深度学习模型基于 GPU 服务器生态进行设计与开发，将这些基于 GPU 服务器生态的框架及模型迁移至

1) https://www.tensorflow.org/

2) https://pytorch.org/

3) https://zh.wikipedia.org/zh/AlphaZero

4) https://e.huawei.com/en/products/cloud-computing-dc/atlas/ascend-910

5) www.aliyun.cn/daily-act/ecs/npu_sales

6) https://cloud.baidu.com/product/kunlun.html

7) 出自《昇腾 AI 处理器及 CANN 软件栈基础》

国产深度学习生态是国产 AI 生态软硬件一体化的必经之路，而昇腾平台为此建立了良好的软硬件基础[8]。

对于 AI 从业者，国产深度学习芯片资源获取困难，如何便利地获取资源是国产平台推广的重大难题。为加快工作者和开发者的迁移进度，昇腾平台展开模型众筹计划。具体而言，昇腾面向 AI 行业科研工作者，提供基于昇腾 AI 处理器的高度集成化行业 SDK 和深度学习训练推理环境，在使用相近的软件框架的前提下，实现不同硬件生态上模型的无缝跨接，这是模型迁移工作中最重要的挑战之一。众筹已完成的深度学习模型案例中的大部分是计算机视觉(CV)相关模型，而自然语言处理模型，如循环神经网络(RNN)等，尚没有实际的迁移案例。为了弥补国产昇腾芯片在自然语言处理算法上的空白、挖掘潜在的架构设计漏洞，本文率先从算法角度切入，迁移自然语言处理四大代表性任务的典型模型至昇腾平台：1)机器翻译生成式任务，对应模型为 RNNSearch[9]；2)机器阅读理解任务，对应模型为 ALBERT[10]；3)序列标注任务，对应模型为 BERT-CRF[1]；4)文本分类任务，对应模型为 ALBERT[10]和 TextING[11]。以上几大案例旨在揭示当前国产深度学习生态与 GPU 生态在自然语言处理模型上的精度差异、性能差异以及国产 AI 平台发展过程中所面临的主要技术挑战及解决方案[1)]。

## 2 待迁移模型

为了验证华为昇腾系统是否支持自然语言处理领域的常用算法模型以及发现潜在缺陷并进行针对性优化，本文选用自然语言处理领域的代表性模型 RNNSearch[9]，ALBERT[10]，BERT-CRF[1]和 TextING[11]作为模型迁移实验案例，迁移目标涵盖神经机器翻译任务、神经阅读理解任务、序列标注任务和文本分类任务，建模方式涵盖生成式建模和判别式建模。

### 2.1 RNNSearch

RNNSearch 是经典的端到端神经机器翻译模型。在建模上，RNNSearch 采用语言建模中具有代表性的类 RNN 模块和注意力模块；在实现上，RNNSearch 有统一且权威的开源实现。出于对代表性和权威性的综合考虑，本文优先选择 RNNSearch 作为自然语言模型迁移的案例。

RNNSearch 出现前，对于不定长的句子，多数 Seq2Seq 翻译模型(如 RNN Encoder-Decoder[12]，LSTM Encoder-Decoder[13]等)倾向于笼统地将其编码成一个定长的、静态的隐藏状态向量，作为后续解码的起始状态。这个操作无视源语言句子的长短，一方面限制了源语言特征的表征能力，导致很多源语言句子语义细节无法得到表征，另一方面迫使每步解码操作都被迫关注到源语言的所有单词。如图 2 所示，为了解决这个问题，RNNSearch 在编码端采用双向的 GRU[14]模块编码源语言文本，保留源语言上下文特征序列；其次，RNNSearch 在解码过程中引入注意力机制，基于当前隐藏状态与源语言上下文表示，计算动态的特征加权权重及上下文向量。RNNSearch 的目标为最大化标准译文的生成概率，对应的目标函数为：

$$L_{\text{RNNSearch}}=-\frac{1}{n}\sum_{t=1}^{n}y_t\log(P(y_t\mid\vec{X},\overleftarrow{X},y_{i<t})) \tag{1}$$

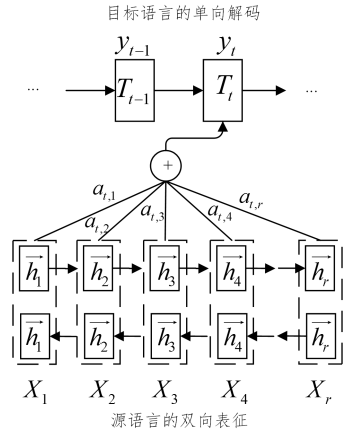


图 2 RNNSearch 的单步解码流程

Fig. 2 Single-step decoding of RNNSearch

### 2.2 ALBERT

近年来，基于 Transformer 拓展的大规模预训练语言模型快速涌现，使得自然语言处理各个领域的模型性能得到了极大的提升，但是大规模预训练所需的参数量和数据量不断扩大，导致预训练难度大大增加。因此，迁移预训练模型是不可忽视的环节，本文选择预训练模型中最具代表性的 BERT 系列作为迁移对象，以探究国产 AI 平台的稳定性。其次，考虑到在实际部署深度学习模型时，小型离线终端的硬件条件限制了模型的规模，因此本文优先选择迁移 BERT 系列中的参数缩减版模型 ALBERT。

ALBERT 是基于 BERT[1]改进的预训练语言模型，它采用权重共享机制，不同的 Transformer Block 之间共享相同的权重，该机制一方面减少了模型参数规模，另一方面有益于稳定模型的训练梯度。此外，ALBERT 将嵌入词向量矩阵分解为两个小矩阵相乘，进一步缩减模型参数量。为了弥补模型参数规模减小导致的性能损失，ALBERT 提出 SOP(Sentence Order Prediction)训练目标，相比 BERT 的训练目标，SOP 的难度更大，这迫使 ALBERT 学习到更深层的语义。

本文将预训练的 ALBERT 运用于机器阅读任务和文本情感分类任务。机器阅读理解任务要求模型根据给定文本和问题，自动预测答案片段在给定文本中的开始位置和结束位置。具体而言，在微调阶段计算模型损失：将机器阅读理解标准答案在文档中的开始位置和结束位置分别记为 $y_t^1$ 和 $y_t^2$。ALBERT 模型预测的开始位置的概率和结束位置的概率分别为 $p_t^1$ 和 $p_t^2$，对应的目标函数为：

$$L_{\text{Squad}}=-\frac{1}{n}\sum_{t=1}^{n}y_t^1\log p_t^1+y_t^2\log p_t^2 \tag{2}$$

在推理阶段，首先，ALBERT 根据模型预测的答案起始位置概率选取前 $k$ 个候选位置；然后，基于起始候选位置，再各自选出前 $k$ 个候选答案结束位置；最后，从候选的起始-结束位置组合中选出得分最高的组合作为 ALBERT 预测的最终答案。

1) NNSearch 工程链接 z：[源代码][模型调优记录]；ALBERT 工程链接：[源代码][模型调优记录]

文本情感分类任务要求模型根据给定文本预测文本情感极性。在微调阶段，将文本的情感标签和模型预测的概率分别记为$y_t$和$p_t$，对应的目标函数为：

$$L_{\text{sst-2}}=-\frac{1}{n}\sum_{t=1}^{n}y_t\log p_t+(1-y_t)\log(1-p_t) \tag{3}$$

### 2.3 BERT-CRF

BERT-CRF 基于 BERT[1] 和条件随机场（CRF）建模，将最后一层隐变量映射至目标空间，并利用 CRF 的转移矩阵计算分数。本文将 BERT-CRF 运用于命名实体识别任务，该任务要求模型根据给定文本，自动抽取特定的实体，如人名、地名等。具体而言，在微调阶段计算模型损失，将真实路径分数和所有路径分数分别记为$S_r$和$S_i$，对应的目标函数为：

$$L_{\text{BERT-NER}}=-\frac{1}{n}\sum_{t=1}^{n}(\log e^{S_r}-\log\sum_{i=1}^{T}e^{S_i}) \tag{4}$$

在推理阶段，BERT-CRF 基于维特比算法对所有路径计算整体概率最大的一组序列，并将其作为最终答案。

### 2.4 TextING

为了涵盖更广的模型类型，提升迁移工作的普适性，本文进一步补充了图神经网络（GNN）类自然语言处理模型，迁移结合 GNN 的文本分类模型 TextING[11]。TextING 待优化的目标函数与 ALBERT 文本情感分类任务相同。

## 3 模型迁移

### 3.1 模型迁移的基本流程

模型迁移的流程就是将业界主流的深度学习框架的原始模型迁移至昇腾 AI 处理器上，它包括 4 个基本步骤：整网功能打通，训练收敛，固化模型，离线推理。整网打通中，在保证模型整网下沉到昇腾 AI 加速芯片 NPU 的同时，开发人员需要保证训练模型最终在昇腾 NPU 芯片上与在市场已有的 GPU（Tesla V100）上取得一致的精度。昇腾平台从硬件支持、软件构架到应用使能上，均和传统的 GPU 服务器平台存在较大差异，因此整网打通阶段是模型迁移过程中出现问题和挑战最多的阶段。

固化模型和离线推理需要将训练后的模型通过昇腾 CANN 异构计算构架，并加速在具体 AI 应用和业务上的效率。离线推理主要使模型在脱离深度学习框架下具有相同的推理能力。在离线模型生成中，首先解析出不同框架下的原始模型的结构和参数，并通过中间的计算图重新定义网络结构；然后，对模型进行量化，压缩模型参数，以及对算子进行编译融合优化；最后，将算子聚合连接编译成为离线模型结构。离线推理模型不但可以达到在线推理的效果，还减少了模型的参数数量，加快了模型的运行速度，提升了昇腾 AI 芯片调用和执行的效率，为小型化设备部署深度学习模型提供了可能性。

### 3.2 自然语言处理模型迁移存在的挑战与解决方案

昇腾平台软硬件架构设计与市场现有方案存在差异，导致迁移包括自然语言处理领域在内的原生态模型到昇腾平台时常常伴随着多种不同类型的问题与挑战。本文以 RNNSearch，ALBERT，BERT-CRF 和 TextING 的模型迁移为典型案例，系统性地总结昇腾平台整体设计上不可遗漏的重要特性，为自然语言处理领域的研究者与开发者提供具有普适性的问题定位方法与解决方案。

具体而言，RNNSearch 的文本解码和编码模块都采用递归式计算图，从而实现生成式解码；而原始 ALBERT 使用固定的文本输入序列，其预训练任务属于定长输出（遮盖单词的预测）的判别式任务，其下游问答任务也属于固定选项数目的判别式任务。两者在实际建模中各具特点，因此两者涉及不同类型的迁移问题。RNNSearch 面临的问题主要包括资源算子不兼容、解码显存空间无法动态注册等；ALBERT 调优任务不涉及动态显存分配问题，在调优过程中，本文更多地侧重于分析昇腾的混合精度在加速运算的同时所面临的精度损失等问题。

#### 3.2.1 设备侧无法为计算图分配动态显存

1）挑战

在昇腾 AI 910 平台上基于 Tensorflow 将模型打通的目标，首先要控制计算图构建拓扑顺序，再依次将算子 Kernel 下沉到加速显卡硬件上完成调度。下沉过程中，开发者需要考虑不同类型的算法模型的显存占用模式存在的差异。为了方便算法研究人员将模型移植到昇腾平台以加快算法的迭代速度，昇腾开发者在运算加速硬件的上层架构设计上需要统筹主流的运算模式。然而，在翻译模型 RNNSearch 迁移案例中，CANN 设计暴露出了一个严重的系统性缺陷：CANN 缺乏为计算图动态分配节点空间的特性。这导致 CANN 无法无缝兼容许多自然语言算法模型，因为序列是自然语言数据的基础数据形式，也是自然语言处理算法建模的基础，序列的长度往往不固定，意味着语言序列张量的尺寸往往是动态的。序列的不定长特性导致序列模型难以正常完成训练流程或者多样本测试，并使研究者在 CANN 架构上迁移或研制自然语言处理模型（尤其是生成式模型）的难度上升。平台的兼容性和易用性大打折扣，这将极大限制自然语言处理领域模型在昇腾平台上的探索和研究。

2）解决方案

由于完善“计算图动态分配节点空间”所需工程量大，CANN 开发经过一定更新迭代周期才能满足，因此，本文借助架构已有的特性来规避此问题。为适配昇腾显卡而设计的 CANN 架构已满足图节点的静态调用的分配模式，能够实现线性分类模型、典型的视觉检测骨干模型 ResNet[15] 等一系列基础模型的完整训练流程。这些模型具有以下特征：具有固定大小的输入输出，不同的批样本不需要调用动态的显存空间，只需要进行初始图下沉到硬件时预分配好固定的空间大小即可，训练过程不需要分配额外的空间。例如，ResNet 输入端数据为固定尺寸的批量图像，尺寸控制为 $B*C*W*H$（$B$ 为批次大小，$C$ 为通道数量，$W$ 为张量宽度，$H$ 为张量长度），模型计算图下沉到加速平台时只需要完成节点注册与固定空间分配即可完成所有数据的训练。

#### 3.2.2 重要的资源算子的兼容性不足

1）挑战

通用的自然语言模型下沉到 Device 侧（昇腾平台中，Device 侧往往指加速运算芯片）所产生的计算图，除包含模型计算图外，还包含文本数据预处理管道。前者涉及包括视觉

模型、自然语言处理模型、推荐系统等在内的通用场景需求，因此前者在架构设计中优先级最高，而后者的开发可能被选择性忽略。

然而，后者的开发程度与语言模型息息相关，它关系到AI平台是否易用。以RNNSearch为例，其预处理计算图采用了面向自然语言处理模型的常见管道，涉及文本读取、噪声文本过滤、构造词表、词表查找、基于批数据的内部序列补齐等全面的文本预处理流程，包含了自然语言处理的常见基础流程。另一方面，Tensorflow和Pytorch等框架对资源算子的下沉已发展得比较成熟，算子支持全面，如果昇腾平台无法有效解决下沉基础资源类算子问题，则会影响研究者的建模体验和研究进度。

2)解决方案

在RNNSearch迁移实验中，Device侧资源算子受阻，暴露出昇腾加速硬件无法下沉资源算子计算图的问题。根据定位知，受阻的原因是昇腾硬件与其CANN架构不完全支持哈希表类资源算子，使得预处理计算图的词表处理节点无法在计算图中打通。为了应对这个难题，本文采用的解决方案如下：将数据预处理管道独立到Host侧(即昇腾CPU)进行重构，使其脱离计算图。需要注意的是，该方案徒然增加了额外的代码调试和时间成本。从自然语言处理领域研究者的角度出发，不完善的资源算子不利于昇腾创建一个科研友好型AI平台。

3.2.3 图算融合

1)挑战

除了考量模型重构的复杂度之外，模型迁移还需要考虑昇腾平台上模型计算效率的优化问题。优化深度学习模型尤其是序列模型，常常伴随频繁的张量切分与拼接，从而产生大量的小算子。计算图中大量的小算子会造成中间计算缓存在硬件的全局显存内，随着拓扑序频繁地下发与调度，运算成本显著增加。基础算子诸如内存拷贝、张量切分、张量合并等被直接转移到昇腾平台时，计算效率大打折扣。受此影响，RNNSearch在迁移过程中遭遇了性能托赘现象。为了处理小算子托赘问题，算法研究者尚无法直击算法的调度开发接口，面向外部研究者的环节只有上层开发接口，即利用给定的算子组合成模型，难以触及底层小算子的分布情况。因此，昇腾平台需要提供具有自适应性的小算子优化解决方案。

2)解决方案

为了更大程度地发挥昇腾平台的高性能运算潜力，昇腾CANN架构提出采用Scope自动融合计算图技术(也称为“图算融合+Auto Kernel Generator”技术)，对某些算子组合情况进行小算子融合。具体而言，昇腾CANN架构采用计算图化简、拆解、合并和特殊化编译等方法，通过分析算子运算关系来实现计算融合，使得计算中间数据直接在共享内存中初始化并计算，以减少对全局内存的读写频数，并充分利用硬件加速资源。图3给出了图算技术如何自动将多层独立注册的残差算子融合成单个Residual Block算子。然而，迄今为止，对于复杂的算子组合形式或者跨域(Scope)的建模场景，研究者只能协同开发人员手动设计融合过程，甚至需要调用底层cuDNN库函数进行优化，再为建模提供解决方案。昇腾的自动图算融合技术尚无法一键式迁移，需要人为干预，还有改进空间。例如，在RNN类中重复利用同一RNN小算子的自动图算融合优化依然不理想，耗时较长，开发者目前仅能建议研究者在构建模型时规避资源调度、拷贝等环节，一定程度上阻止计算性能下降，但尚不能完全解决性能问题。因此，国产昇腾平台对于生成式模型的图算优化有待加强，具体分析详见4.3.2节。

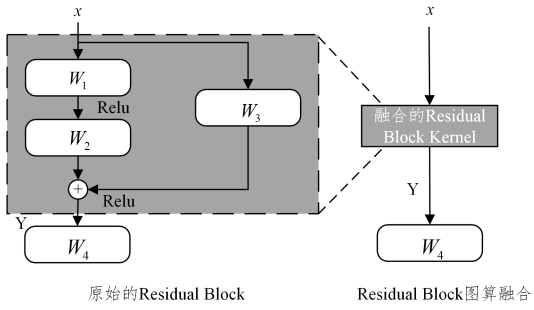


图3 昇腾计算图中的算子融合例子

Fig. 3 Operator fusion example in Ascend's computation

3.2.4 混合精度训练导致精度损失

1)挑战

为方便研究者和开发人员基于昇腾910平台快速构建国产AI应用，昇腾910平台正在迁移包括Tensorflow和Pytorch在内的业界主流深度学习框架。由于平台的硬件基础不同以及开发过程的不完善，部分算子难免存在精度偏误。这些偏误一方面来源于算子实现方式的差异，另一方面来源于昇腾CANN秉持的“混合精度”架构特性。主流的深度学习系统为了稳定训练，往往倾向于使用统一的精度表示运算结果。为了减少内存占用和缩短模型训练时间，昇腾CANN架构混合半精度Float16与单精度Float32方式来选择性地加速计算过程。在已有的混合精度研究中，Micikevicius等[16]针对网络中单精度类型的算子，按照优化策略，将部分算子降低为半精度，同时结合Loss Scaling，弥补降低精度带来的精度损失。需要注意的是，迁移成混合精度模型时，显式数据类型转换容易导致算子结果出现误差。例如，按照开源格式，与BERT同源的ALBERT建模中的多头注意力机制的掩码往往被显式强制转换为FP32格式并与极小数相乘；进而，在多头注意力机制模块中，该乘积与FP32或者FP16类型的logit张量进行相加。当数据类型转换在昇腾NPU执行时，该加减运算直接产生了过大误差，导致ALBERT的前向计算图精度不足。具体而言，昇腾平台上ALBERT注意力机制被修正前后，与基于GPU训练的ALBERT标准实现相比，运算结果的相似度分别为0.50和0.97，可见混合精度模式容易造成严重的精度误差传递。因此，为了防止精度损失，混合精度模型在很大程度上对显式数据类型转换有所限制，约束了研究者的建模自由。

2)解决方案

为了应对精度损失问题，昇腾为开发者提供了跨硬件的精度调试方案，以查明精度误差的所在位置，这个过程的通用调试粒度为算子级别。总结起来，混合精度训练导致的精度损失，可能来自CANN算子支持不善、参数数据类型转换等

原因。ALBERT 调试过程中，研究者需要将昇腾平台上网络下沉的算子计算结果，同标准的基于 GPU/CPU 的算子计算结果进行精度对比，从而定位出第一个出现精度误差的算子，即对掩码进行的数据类型转换 Cast。昇腾平台常用的基于算子计算结果、计算方法对比包含余弦相似度、最大绝对误差等。当 ALBERT 模型中存在将下沉到 NPU 中的半精度算子强制类型转换为单精度算子的运算时，会导致算子和 GPU/CPU 上算子余弦相似度的结果存在误差，这些误差会随着网络的传播逐渐放大，导致模型最后的计算结果存在较大偏差。本文采用的解决方案如下：将 ALBERT 第一个出现的精度误差的算子操作移动到计算图开始，使得掩码在进入多头注意力机制前完成数据类型转换。此外，当模型显式调用数据类型转换以达成手动混合精度时，无需昇腾平台的混合精度。

## 4 实验与分析

### 4.1 数据集

为测试迁移至昇腾 AI 芯片后 RNNSearch，ALBERT，BERT-CRF 和 TextING 模型的有效性，本文分别在机器翻译、机器阅读理解、文本分类和命名实体识别多个任务上开展了验证实验。对于神经机器翻译任务，本文基于 IWSLT 2015 英德训练数据集[1)]进行实验分析并以 IWSLT 2015 英德测试集测试翻译性能。对于机器阅读理解任务，本文主要在 SQuAD 2.0[17]数据集上进行实验分析。对于文本情感分类任务，本文主要在 SST-2[18]上进行实验，以上两种数据集均测试迁移后 ALBERT-base 和 ALBERT-large 两种规格的模型性能。另外，本文使用电影评论文本情感分类数据集 MR[19]，基于 TextING 探究了 GNN 与 NLP 相结合的模型优化方法。对于命名实体识别任务，本文主要在 CoNLL-2003[20]数据集上进行实验并测试迁移后 BERT-CRF 上的性能。表 1 列出了所使用的训练数据和测试数据的统计信息。

表 1 迁移实验语料规模统计

Table 1 Statistics of datasets in migration experiment

| 数据集 | 类型 | 数量 |
|---|---|---|
| IWSLT 2015 英德数据集 | 训练集 | 5 852 458 |
| | 验证集 | 2 169 |
| SQuAD 2.0 | 训练集 | 130 319 |
| | 验证集 | 11 873 |
| SST-2 | 训练集 | 25 137 |
| | 验证集 | 1 389 |
| CoNLL-2003 | 训练集 | 14 987 |
| | 验证集 | 3 466 |
| | 测试集 | 3 684 |
| MR | 训练集 | 6 398 |
| | 验证集 | 710 |
| | 测试集 | 3 554 |

### 4.2 实验设置

#### 4.2.1 超参数选择与训练方法

由于 RNNSearch 的原始作者没有发布开源代码，因此本文 RNNSearch 模型迁移实验参考清华发布的开源工程[2)]作为 RNNSearch 的标准实现；ALBERT 模型迁移实验以谷歌发布的预训练模型 ALBERT-2.0 为基础，在 SQuAD 2.0 和 SST-2 上分别进行微调并将其作为基线模型；优化器均选用 AdamW 优化器。BERT-CRF 和 TextING 采用开放的源工程作为标准实现；优化器均选用 AdamW 优化器 RNNSearch。ALBERT 和 BERT-CRF 迁移实验使用的学习率策略均严格按照原始代码的参数设定。实验中，受限于显存大小，ALBERT-base 和 BERT-CRF 的满载批尺寸为 32，ALBERT-large 的满载批尺寸为 16。特别地，出于简化矩阵计算与加速运算的考虑，CANN 架构结合硬件底层设计，以精度损失换取速度提升，使用 Nvidia Tensor Core FP16 接口优化 MulMatV2，ConvV2 等常见网络基础结构，并针对其输入输出定义了半精度格式限制，以至于昇腾平台训练流程往往需要开启混合精度训练模式。为了保证本文实验结论的通用性，RNNSearch 和 ALBERT 的训练流程均开启混合精度设定。为了规避该设定所产生的精度损失，RNNSearch 和 TextING 使用 Loss Scaling 技术弥补梯度上的误差。

#### 4.2.2 测评方法

关于测评指标，RNNSearch 使用双语替换测评值[21]（Bi-Lingual Evaluation Understudy，简称 BLEU）作为翻译模型的评价指标；ALBERT 使用模糊匹配度（F1）和精准匹配度（Exact Match，简称 EM）作为评测指标，计算预测结果和标准答案的匹配程度[10]；BERT-CRF 采用准确率（Accuracy）、精准度（Precision）和召回率（recall）作为序列标注性能指标；ALBERT 和 TextING 采用分类准确率衡量文本分类模型的性能。

关于精度基线在不同平台上的横向对比，本文优先考虑公开的模型实验报告。首先，原始 RNNSearch 未公开 IWLST 2015 英德赛道技术报告，而所使用的训练数据的来源和规模往往直接影响 RNNSearch 的翻译精度。为了排除语料的影响，本文在公开的 GPU 上训练的 RNNSearch 版本中选出使用相同训练集且在同一验证集上 BLEU 值最优的模型作为 RNNSearch 的对标基线。其次，ALBERT，BERT-CRF 和 TextING 分别对标原始论文中公布的性能。

关于模型性能在不同平台上的横向对比，本文选取昇腾 NPU 加速卡和 Tesla V100 显卡测试模型在不同批尺寸输入下的平均单步运行时间。

### 4.3 实验结果

#### 4.3.1 模型精度比较

迁移实验成功与否的第一项指标是模型精度是否达标。表 2 列出了 RNNSearch，ALBERT-base，ALBERT-large，BERT-CRF 和 TextING 与基线模型在测试集上的精度。根据表 2 可知，自然语言处理任务的昇腾平台移植复现精度或者昇腾平台离线推理精度均接近或超过了公开的精度。因此，昇腾平台支持的混合精度学习能够在不降低模型精度的情况下，降低模型对内存的要求，提高模型的训练效率。

1) https://workshop2015.iwslt.org/

2) https://github.com/THUNLP-MT/THUMT

表 2 迁移实验在各测试集的精度对比

Table 2 Accuracy results of each testset in migration experiments

| 模型名称 | RNNSearch | | |
|---|---|---|---|
| IWSLT 2015 | BLEU↑ | | |
| 对标模型 | 26.76 | | |
| NPU | 27.25 | | |
| 离线推理 | 27.20 | | |
| 模型名称 | ALBERT-base | | |
| SquAD 2.0 | F1↑ | EM↑ | |
| 对标模型 | 82.10 | 79.30 | |
| NPU | 82.45 | 79.45 | |
| 离线推理 | 82.42 | 79.42 | |
| 模型名称 | ALBERT-large | | |
| SquAD 2.0 | F1↑ | EM↑ | |
| 对标模型 | 84.90 | 81.80 | |
| NPU | 85.50 | 82.53 | |
| 离线推理 | 85.53 | 82.54 | |
| 模型名称 | ALBERT-base | ALBERT-large | |
| SST-2 | Accuracy↑ | | |
| 对标模型 | 92.90 | 94.90 | |
| NPU | 92.38 | 94.15 | |
| 模型名称 | BERT-CRF | | |
| CoNLL-2003 | Accuracy↑ | Precision↑ | Recall↑ |
| 对标模型 | 98.15 | 90.61 | 88.85 |
| NPU | 98.02 | 90.06 | 87.86 |
| 模型名称 | TextING | | |
| MR | Accuracy↑ | | |
| 对标模型 | 79.8 | | |
| NPU | 80.0 | | |

### 4.3.2 模型速度比较

除了精度指标外，本文还对比了机器阅读理解和机器翻译两类典型任务所对应的模型在不同硬件上的平均训练速度，如表 3 所列。在 V100 上，给定不同的批尺寸，ALBERT 训练耗时均为昇腾平台耗时的 2.9 倍以上；随着批尺寸的增大，加速比不断上升，ALBERT-base 和 ALBERT-large 的加速比最高，分别达到最高值 3.9 与 6.4。此外，ALBERT-large 的加速比均高于 ALBERT-base 的加速比，且是 ALBERT-base 的加速比的 1.5～1.8 倍。ALBERT-base 的模型参数规模为 $12\times10^6$，ALBERT-large 的模型参数规模为 $18\times10^6$。实验说明，在合理范围内模型参数规模越大，昇腾平台的训练加速效果越明显。受限于小算子融合不理想、动态显存分配不支持等问题，RNNSearch 训练耗时拖赘严重，无法对标 V100 的速度，详见 4.4.1 节。

表 3 昇腾 910 与 Tesla V100 上 ALBERT 和 RNNSearch 的平均单步训练耗时

Table 3 Average single-step training time of ALBERT and RNNSearch on Ascend 910 and Tesla V100

(ms/step)

| 批尺寸 | ALBERT-base | | | ALBERT-large | | | RNNSearch | | |
|---|---|---|---|---|---|---|---|---|---|
| | V100 | NPU | 倍率 | V100 | NPU | 倍率 | V100 | NPU | 倍率 |
| 32 | 765 | 195 | 3.9 | — | — | — | 166.3 | 485.1 | 0.34 |
| 16 | 402 | 106 | 3.8 | 1 312 | 205 | 6.4 | 133.4 | 369.7 | 0.36 |
| 8 | 214 | 60 | 3.6 | 672 | 119 | 5.6 | 118.6 | 254.5 | 0.47 |
| 4 | 115 | 42 | 2.7 | 355 | 74 | 4.8 | 116.4 | 250.1 | 0.47 |
| 2 | 65 | 29 | 2.2 | 209 | 52 | 4.0 | 110.2 | 210.3 | 0.53 |
| 1 | 39 | 20 | 2.0 | 100 | 34 | 2.9 | 97.6 | 175.0 | 0.56 |

最后，表 4 列出了迁移实验中单个样本的平均推理耗时。推理过程只涉及前向计算图，不包含梯度回传计算图。经过昇腾 910 AI 芯片以及异构计算框架 CANN 优化与算子融合后，ALBERT-base 和 ALBERT-large 两个模型均在不损失精度的影响下大大加快了推理的速度。而 RNNSearch 的耗时拖赘问题在推理期间依然存在。

表 4 模型单样本推理平均耗时

Table 4 Average single-sample inference time of each model

(ms/sample)

| 模型名称 | NPU | V100 |
|---|---|---|
| ALBERT-base | 11.4 | 20.6 |
| ALBERT-large | 15.9 | 49.8 |
| RNNSearch | 179.7 | 90.2 |

## 4.4 分析

### 4.4.1 动态 shape 不兼容造成的延迟

如 3.2.1 节所述，昇腾加速架构暂未开发动态显存分配特性。为规避这个问题，本文在实现 RNNSearch 时采用了将不定长序列强制退化为定长序列的静态显存占用方案：搜索数据集预估文本序列潜在的最大长度，将所有序列补齐为最大长度，得到统一的文本尺寸的伪序列。

虽然该规避思路具有代表性，适合于同样存在动态序列长度问题的语言模型，但是，该方案伴随着速度下降问题，尤其在沿序列维度进行自回归式推演的语言任务中，真实数据集中句子的长度往往呈现长尾分布、中短句子居多，迫使模型按照最长句子序列长度补齐，严重影响训练的前向传播和反向传播效率，时间开销上浮明显。为了衡量 CANN 开发动态显存分配的必要性，本文在 V100 上对比了批尺寸为 32 时 RNNSearch 使用定长序列和变长序列两种设定的单样本平均训练耗时，实验结果如图 5 所示(注：在本实验完成期间，昇腾暂不支持变长序列计算图)。

根据 V100 的实验结果，变长序列的单步耗时是定长序列单步耗时的 65%，这表明不支持动态分配空间的特性极有可能严重降低自然语言处理模型在 NPU 平台上的训练效率。另外，在定长序列实验中，NPU 的平均耗时是 V100 的 2 倍，原因是 RNN 算子涉及太多小算子，目前昇腾的图算融合技术无法自动处理该场景下的小算子，产生了过多算子切换开销。

### 4.4.2 混合精度下的训练稳定性对比

算子在混合精度下传播时，数据类型转换过程存在一定程度的精度损失，连续频繁的前向计算精度损失或者反向梯度精度损失可能造成模型训练崩溃。特别地，对于 RNNSearch 类典型的自回归模型，单步训练时多次重复调用每个 RNN 类模块，梯度回传的链路较长，混合精度的梯度损耗传递明显加剧，导致模型对梯度精度的敏感性上升。针对此问题，本文借鉴 Loss Scaling 技术，缓解混合精度下梯度数据的精度损失。具体而言，Loss Scaling 在更新梯度前对损失函数值进行缩放，算取梯度后再对梯度进行一定比例放缩，用于更新模型参数。除了 Loss Scaling 技术，模型设计中稳定梯度的另一种常见方式是采用梯度裁剪技术(Gradient Clipping)来缓解计算误差导致的梯度上溢。

为了探究更合理的训练策略，本文以 LS 表示 Loss Scaling、GC 表示 Gradient Clipping，以 X-Y 表示在 X 加速平台上

按照 Y 设定训练 RNNSearch，进行了 4 种 RNNSearch 实验：1）V100-FP32-w/o LS w/o GC；2）NPU-mixed-w/o LS w/o GC；3）NPU-mixed-w/o LS w/ GC；4）NPU-mixed-w/ LS w/o GC。

其中，原始的 RNNSearch 设定是 V100-FP32-w/o LS w/o GC，即在非国产的标准 GPU 上进行单精度训练，该设定不执行混合精度训练。另外，NPU-mixed-w/o LS w/o GC 的梯度不做任何处理。

图 4 是上述 4 种实验的损失函数变化图。混合精度下不对梯度做任何处理的 RNNSearch 训练稳定性很低，混合精度的梯度值与精度损失呈正相关，因此损失函数的下降使得 RNNSearch 的梯度精度相对损失率随之上升，梯度不合理更新，最终损失函数在训练前期就快速溢出，无法正常训练。而 NPU-mixed-w/ LS w/o GC 解决了这个问题，呈现平稳的损失函数变化曲线；在 RNNSearch 训练中后期，NPU-mixed-w/ LS w/o GC 的损失函数比 V100-FP32-w/o LS w/o GC 收敛到了更低的水平。

然而，梯度裁剪没有帮助 RNNSearch 实现鲁棒的训练。在训练的后期，梯度依然不可控，导致损失函数再次出现上溢现象。以上实验说明，Loss Scaling 技术比梯度裁剪技术更适合作为混合精度下的稳定训练策略。

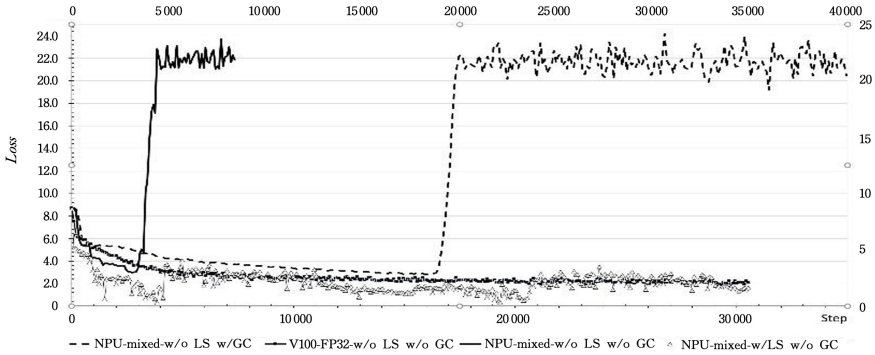


图 4 RNNSearch 在 4 种训练模式设定下损失函数的变化情况

Fig. 4 Loss curve with four training settings of RNNSearch

为了排除偶然因素并进一步验证 Loss Scaling 技术对于提高训练稳定性的普适性，本文结合另一个文本情感分类自然语言处理任务开展了昇腾平台 TextING 模型迁移实验。相比 RNNSearch，TextING 的计算图链路较短，计算图中混合精度的误差传导影响减弱，有利于考察 Loss Scaling 技术的通用性。实验结果如表 5 所列，在 NPU 设备混合精度训练过程中，Loss Scaling 技术一定程度地提升了 TextING 在验证集和测试集上的文本分类准确率。此外，图 5 给出了 TextING 在 Loss Scaling 模式下验证集上的分类准确率变化趋势。

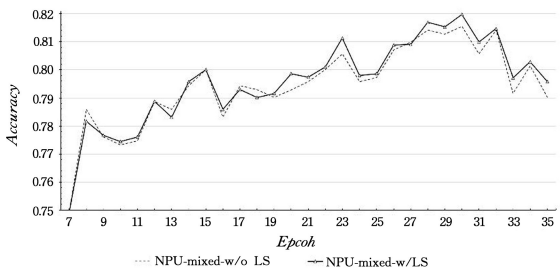


图 5 TextING 在 Loss Scaling 模式下验证集分类准确率曲线

Fig. 5 TextING's accuracy curve on validation set in Loss Scaling mode

一方面，链路较短的 TextING 没有出现梯度爆炸、训练崩溃的现象，符合计算链路短的模型受混合精度影响减弱的假设；另一方面，图 5 表明 Loss Scaling 帮助 TextING 模型在训练的后半期（Epoch 大于 19）整体呈现更优的分类效果，说明 Loss Scaling 不仅有助于解决模型的训练崩溃问题，还有助于提升模型训练的最终效果。

表 5 TextING 基于 MR 数据集的 Loss Scaling 实验记录

Table 5 Loss Scaling experiment results of TextING on MR

| 测试指标 | Validation Accuracy ↑ | Test Accuracy ↑ |
|---|---|---|
| GPU | — | 79.8 |
| NPU w/o LS | 81.8 | 79.3 |
| NPU w/LS | 82.4(+0.6) | 80.0(+0.7) |

在实际操作中，混合精度模式还容易放大潜在的人为不合理运算带来的精度失误。例如，ALBERT 案例所属的阅读理解数据集任务中，其模型的计算图与 TextING 类似，拓扑链相对简短，精度的误差传递程度轻，ALBERT 无需借助 Loss Scaling 技术便足以保持稳定的训练梯度；然而，稳定、不上溢的梯度不能保证模型训练的最终性能，正如 3.2.4 节所述，多头注意力模块中半精度张量与单精度张量的混合计算精度敏感，影响 ALBERT-base 的最终训练效果。表 6 和表 7 列出了 ALBERT-base 的多头注意力模块经过数据类型转换修正前后在测试集上的性能。

表 6 Squad 2.0 混合精度实验记录

Table 6 Mixed precision training results on Squad 2.0

| Squad 2.0 | F1 ↑ | EM ↑ |
|---|---|---|
| 修正前 | 50.40 | 47.21 |
| 修正后 | 82.45 | 79.45 |

表 7 SST-2 混合精度实验记录

Table 7 Mixed precision training results on SST-2

| SST-2 | ALBERT-base Accuracy ↑ | ALBERT-large Accuracy ↑ |
|---|---|---|
| 修正前 | 88.24 | 89.93 |
| 修正后 | 92.38 | 94.15 |

实验结果表明，多头注意力模块中混合精度计算所造成的精度损失导致 ALBERT-base 在 Squad 2.0 数据集上降低了 32.05 个 F1 值和 32.24 个 EM 值，在 SST-2 上降低了 3～4 个 Accuracy 值。

表 8 列出了 BERT-CRF 在 COLL-2003 数据集上的多头注意力模块经数据类型转换修正前后在测试集上的性能，其会导致降低 39.83 个 Accuracy 值，69.9 个 Precision 值和 74.78 个 Recall 值。需要注意的是，对于相同的算法结构，不同的研究者在代码编写上往往存在不同程度的差异，在混合精度模式下微小的代码逻辑差异可能会造成不同程度的精度损失，而修正精度损失的工作往往需要大量的调试成本。因此，混合精度模式要求平台开发人员完善张量类型自动转换，或根据不同的建模细节自动触发关于精度损失的警示，实现鲁棒的张量运算。

得益于晟腾平台的 Scope 自动融合计算图技术和静态现存分配技术，开发人员能动态监测模型训练过程中耗时长和精度低的算子，并进行针对性规避和改进。实验结果表明，迁移后的模型能更好地适应混合精度训练，在不降低模型精度的情况下提升模型训练速度。

表 8 CoNLL-2003 混合精度实验记录

Table 8 Mixed precision training results on CoNLL-2003

| BERT-CRF | Accuracy ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|
| 修正前 | 58.19 | 20.16 | 13.08 |
| 修正后 | 98.02 | 90.06 | 87.86 |

## 5 针对国产深度学习平台研发的建议

以昇腾 AI 平台为代表的国产深度学习平台，已经取得长足的进展，但是在系统架构的设计和研发上，依然存在不少缺陷。本节针对如何更合理地构建国产深度学习平台，从迁移案例出发，面向自然语言处理算法的高性能计算需求，为构建国产人工智能平台总结了以下建议。

### 5.1 易用性

国产 AI 平台的操作难易程度决定了人工智能平台是否可以被大众接受。国产深度学习平台不仅要做到开发友好型，而且应该以研究友好型平台为目标。以昇腾架构的图算融合新特性为例，研究者往往难以在短时间内为复杂的模型自助重构与测试融合效果。为了达到这个目的，昇腾平台需要从底层进一步完善和整合图算融合场景。因此，系统所提出的新特性应强调兼容性，帮助国产 AI 系统参与者实现无缝迁移模型。

### 5.2 统筹平衡

国产深度学习平台的迁移实验应该统筹不同领域的算法。目前，昇腾平台的迁移案例按照计算机视觉、自然语言处理和推荐系统进行分配。而现有尝试中，自然语言处理模型的案例比例远不足计算机视觉案例。这两类模型在建模方式上有显著的差异，迁移与测试的过程需要开发人员统筹与平衡自然语言处理算法的平台实现。比如，架构统筹规划不够全面，导致与 RNNSearch 相似的生成类模型为了适配静态显存分配而牺牲加速卡性能，同时又由于难以避开小算子而进一步增加了训练成本。

### 5.3 透明性

鼓励开源面向研究者一侧的算子底层代码，完善平台教程和技术论坛，将已有迁移案例的经验有序地整合到相关文档中。代码和经验的透明性有利于更多人了解国产深度学习平台的架构，做到有迹可循、有错可查、群策群力共同分析，最终吸引更多开发者参与到国产深度学习系统开发的浪潮中。

**结束语** 针对自然语言处理算法，本文立足于国产昇腾 AI 平台，探究如何更好地构建国产深度学习平台。目前，以昇腾为代表的国产人工智能平台虽然已经具备高性能算例的硬件基础设计，但是面向自然语言处理算法与业务的兼容性依然有待提高。基于 RNNSearch 和 TextING 的迁移实践，重点探究了生成式建模任务所必须面临的动态显存分配特性、资源算子兼容问题与梯度调优方案等挑战；基于 ALBERT 和 BERT-CRF 的迁移实践，本文探究了昇腾平台如何平衡精度和速度。最后，本文为促进国产深度学习平台的自然语言处理生态良好发展总结了经验，为国产人工智能系统的自然语言处理算法建设提出了建议，并为国产自然语言处理算法软硬件平台的开发提供了早期模板。

## 参 考 文 献


[1] LING S, NGUYEN K, ROUX-LANGLOIS A, et al. A lattice-based group signature scheme with verifier-local revocation [J]. Theoretical Computer Science, 2018, 730(19): 1-20.

[2] DEVLIN J, CHANG M W, LEE K, et al. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding [C]// Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics. Human Language Technologies, Volume 1, Minneapolis, 2019: 4171-4186.

[3] JIANG C, CHEN T S. Chinese AI starts from "core"[J]. Zhong Guan Cun, 2019(2): 48-51.

[4] Anonymous. Huawei Released AI Processor Ascend 910[J]. Office Information, 2019, 24(19): 25.

[5] RAN Y L. AI Chip Industry And Trends [J]. Big Data Time, 2019(4): 40-45.

[6] Anonymous. Alibaba Released Self-Developed AI Chip Han-Guang 800[J]. Intelligent Building & Smart City, 2019(10): 5.

[7] MA Y J, YU D H, WU T, et al. PaddlePaddle: An Open-Source Deep Learning Platform From Industrial Practice[J]. Frontiers of Data and Domputing, 2019, 1(1): 105-115.

[8] YU F. Research on the Next-Generation Deep Learning Framework[J]. Big Data Research, 2020, 6(4): 69-78.

[9] BAHDANAU D, CHO K, BENGIO Y. Neural Machine Translation by Jointly Learning to Align And Translate[C]// Proceedings of the International Conference on Learning Representations, 2015.

[10] LAN Z Z, CHEN M, GOODMAN S, et al. ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations[C] // Proceedings of the International Conference on Learning Representations. Addis Ababa, 2020: 1-17.

[11] ZHANG Y F, YU X L, CUI Z Y, et al. Every Document Owns Its Structure: Inductive Text Classification via Graph Neural

Networks[C]// Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020：334-339.

[12] SUTSKEVER I, VINYALS O, LE Q. Sequence to Sequence Learning with Neural Networks[C]// Proceedings of Advances in Neural Information Processing Systems. Cambridge, 2014：3104-3112.

[13] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural Computation, 1997, 9(8)：1735-1780.

[14] CHUNG J Y, GÜLCEHRE C, CHO K, et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling [C]// Proceedings of Advances in Neural Information Processing Systems Deep Learning and Representation Learning Workshop. 2014.

[15] HE K M, ZHANG X Y, REN S Q, et al. Deep Residual Learning for Image Recognition[C]// Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, 2016：770-778.

[16] MICIKEVICIUS P, NARANG S, ALBEN J, et al. Mixed precision training[C]// Proceedings of International Conference on Learning Representations. Vancouver, 2018.

[17] RAJPURKAR P, JIA R, LIANG P. Know What You Don't Know：Unanswerable Questions for SQuAD[C]// Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, 2018：784-789.

[18] WANG A, SINGH A, MICHAEL J, et al. GLUE：A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding[C]// Proceedings of the 2018 EMNLP Workshop：Analyzing and Interpreting Neural Networks for NLP. Brussels, 2018：353-355.

[19] PANG B, LEE L. A Sentimental Education：Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts [C]// Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics. 2004：271-278.

[20] SANG E F T K, MEULDER F D. Introduction to the CoNLL-2003 Shared Task：Language-Independent Named Entity Recognition[C]// Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL. 2003：142-147.

[21] PAPINENI K, ROUKOS S, WARD T, et al. Bleu：A Method for Automatic Evaluation of Machine Translation[C]// Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2002：311-318.

**GE Huibin**, born in 1997, postgraduate. His main research interests include deep learning and nature language processing.

**XIONG Deyi**, born in 1979, Ph.D, professor, Ph.D supervisor. His main research interests include natural language processing and machine translation.

（责任编辑：喻藜）

# CCF YOCSEF 换届选举完成，沈华伟当选新一任 AC 主席

2023 年 12 月 16 日下午，CCF YOCSEF 第二十六届学术委员会第二次会议在奇安信安全中心一层报告厅举行，会议进行了 AC 委员的选举以及学术委员会主席会议成员的换届选举。

经过差额竞选和无记名投票，王宏宁（清华大学）、张宇超（北京邮电大学）当选为学术界 AC 委员；刘伟（小米公司）当选为非学术界 AC 委员。AC 委员的选举由 YOCSEF 现任主席高志鹏主持。

YOCSEF 现任 AC 委员、中国科学院计算技术研究所沈华伟当选新一届 YOCSEF（2024－2025）学术委员会主席。中国人民大学范举、深圳大学陈小军、中国科学院信息工程研究所于静当选 AC 副主席。微软亚洲研究院陈昊、华北电力大学张莹当选学术秘书。主席会议成员的换届选举由 YOCSEF 秘书长谭晓生主持。

AC 委员的选举结果已通过 CCF YOCSEF 指导委员会批准；新一届 CCF YOCSEF 学术委员会主席会议成员的选举结果经 YOCSEF 主席和秘书长共同确认后，已通过 CCF 秘书长批准，他们将于 2024 年 5 月上任（AC 委员任期三年，主席会议成员任期一年）。

**据 CCF 微信公众号**