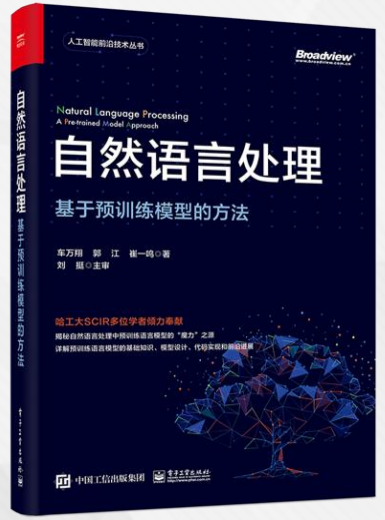


基础工具集与常用数据集

车万翔、郭江、崔一鸣

社会计算与信息检索研究中心
哈尔滨工业大学

基础工具集与常用数据

基础工具集

- NLTK
- LTP
- PyTorch

常用数据集

- Wikipedia
- Common Crawl

NLTK

- Natural Language Toolkit
  - https://www.nltk.org/
- 多种语料库和词典资源
  - 生文本、PennTreebank样例
  - WordNet
- 基本的自然语言处理工具集
  - 分句
  - 标记解析
  - 词性标注
  - 句法分析
- 更多英文自然语言处理工具集
  - CoreNLP、spaCy等

https://data-flair.training/blogs/nltk-python-tutorial/

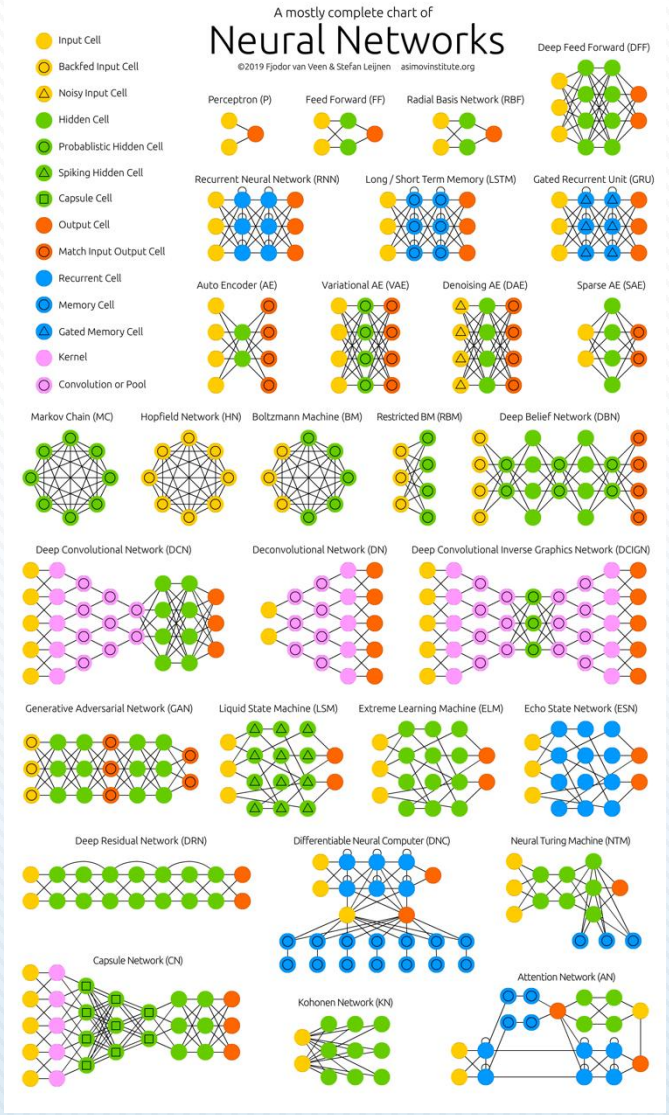
- 语言技术平台（Language Technology Platform，LTP）
  - http://ltp.ai
  - 高效、高精度中文自然语言处理基础平台
  - 中文词法、句法、语义分析等6项自然语言处理核心技术
- 2020年7月发布4.0版本
  - 基于预训练模型
  - 多任务学习机制
- 安装
  - $ pip install ltp

LTP调用示例

```
>>> sentences = ltp.sent_split(["南京市长江大桥。", "汤姆生病了。他去了医院。"
    ]) # 分句
>>> print(sentences)
['南京市长江大桥。', '汤姆生病了。', '他去了医院。']
>>> segment, hidden = ltp.seg(sentences)
>>> print(segment)
[['南京市', '长江', '大桥', '。'], ['汤姆', '生病', '了', '。'], ['他', '去', '
    了', '医院', '。']]
>>> pos_tags = ltp.pos(hidden) # 词性标注
>>> print(pos_tags) # 词性标注的结果为每个词所对应的词性，LTP使用的词性标记集与
    # NLTK不尽相同，但基本大同小异
[['ns', 'ns', 'n', 'wp'], ['nh', 'v', 'u', 'wp'], ['r', 'v', 'u', 'n', 'wp']]
```

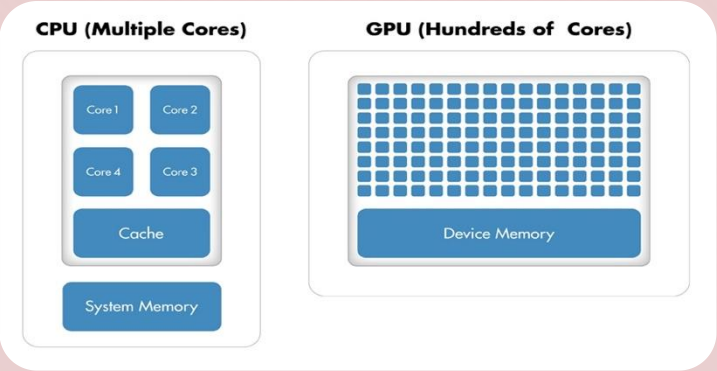
PyTorch

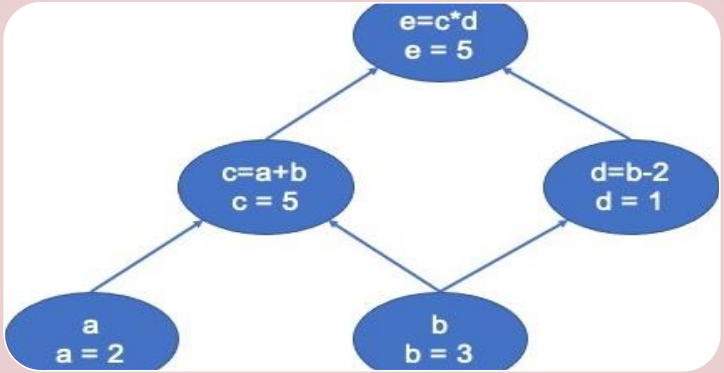
- Facebook开发的开源深度学习库
  - https://pytorch.org/
- 基于张量（Tensor）的数学运算工具包
  - GPU加速的张量计算
  - 自动进行微分计算
- PyTorch的优势
  - 框架简洁
  - 入门简单，容易上手
  - 支持动态神经网络构建
  - 与Python语言无缝结合
  - Debug方便

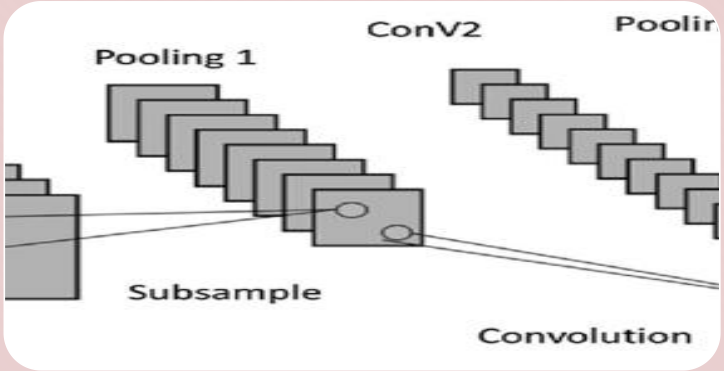


https://www.asimovinstitute.org/neural-network-zoo/

PyTorch的三要素







张量（Tensor）

- 数据存储
- 支持GPU

表达式（Expression）

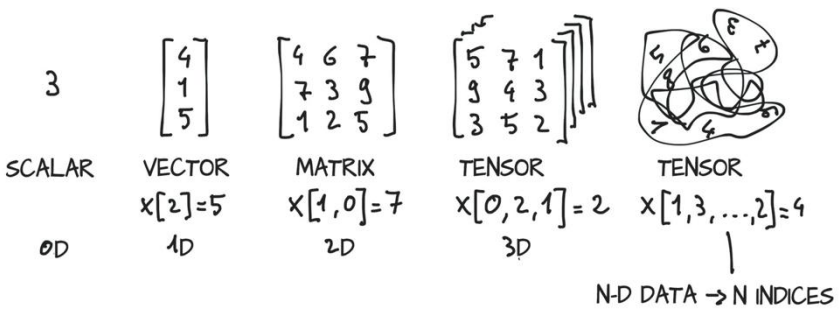
- 数据运算
- 自动微分计算

模块（Module）

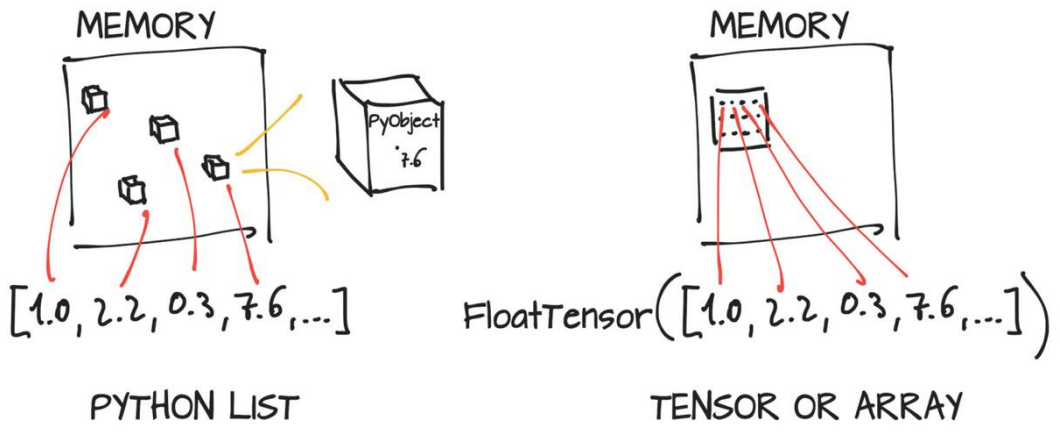
- 神经网络层
- 支持自定义

张量（Tensor）

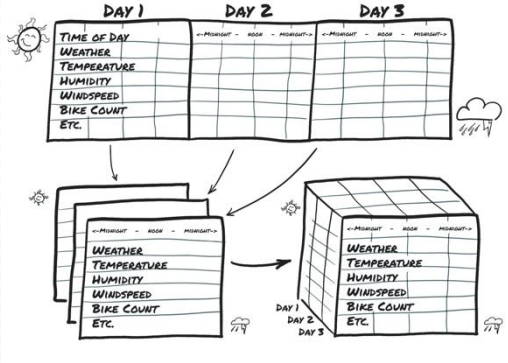
- 又称为多维数组



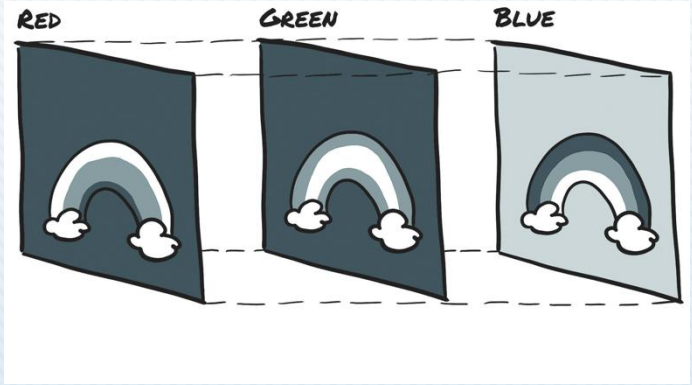
- 与Python列表的区别



- 可表示现实世界中的各种数据
  - 表格、时间序列



  - （彩色）图像



batch = torch.zeros(100, 3, 256, 256, dtype=torch.uint8)

PyTorch作为张量库

- 支持各种张量操作（类似NumPy）
  - 创建张量
  - 索引、切片
- 支持GPU
  - 快！

$M * M * M$ $M \in \mathbb{R}^{1000\times1000}$

Numpy

```
In [2]: M = numpy.random.randn(1000,1000)

In [3]: timeit -n 500 M.dot(M).dot(M)
500 loops, best of 3: 30.7 ms per loop
```

PyTorch

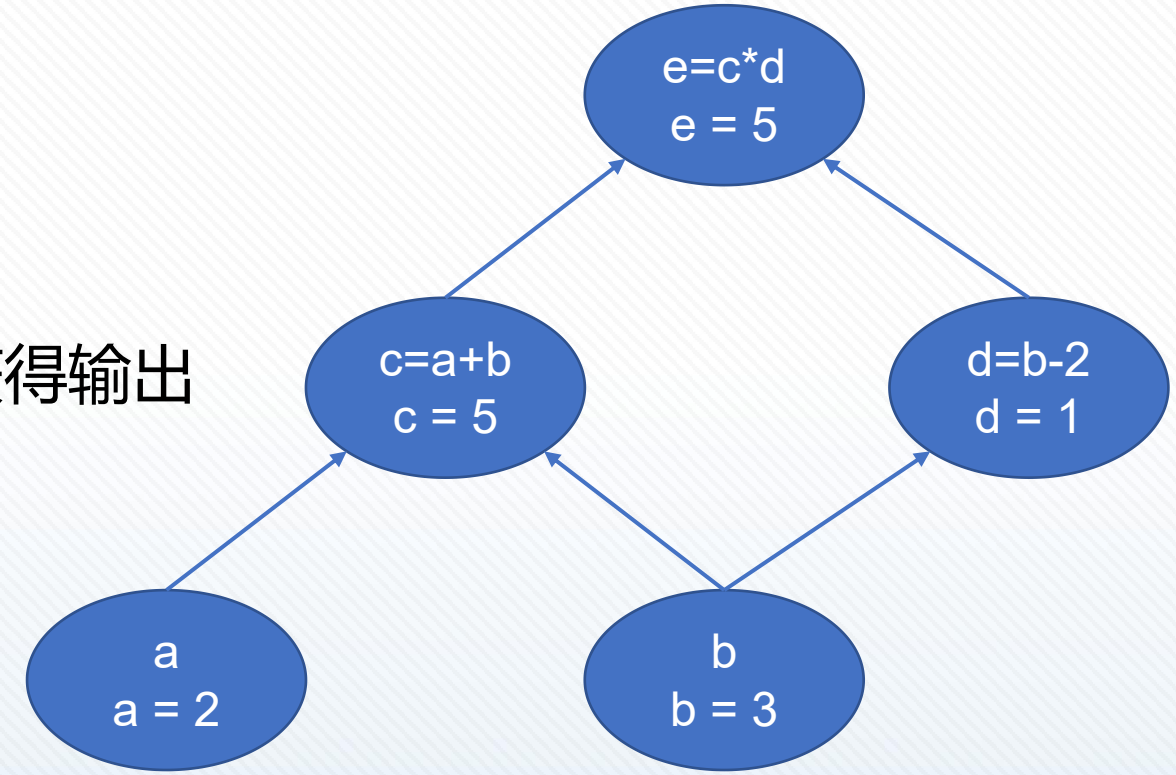
```
In [4]: N = torch.randn(1000,1000).cuda()

In [5]: timeit -n 500 N.mm(N).mm(N)
500 loops, best of 3: 474 µs per loop
```

表达式（Expession）

- 通过计算图描述表达式
  - 如：e = (a + b) * (b - 2)
- 计算图中每个节点执行一个运算
  - 前向运算（Forward）：根据输入获得输出

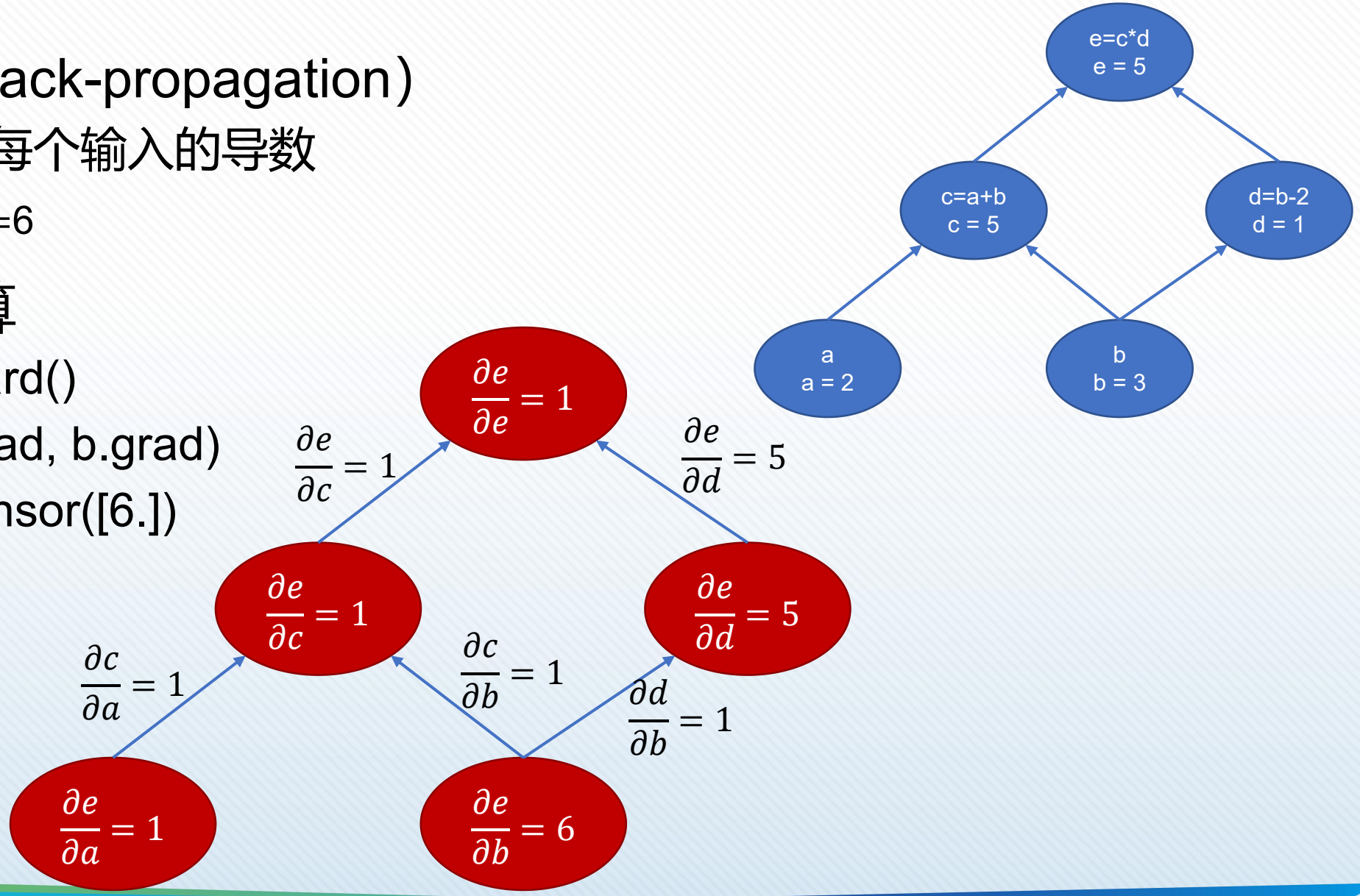
```
>>> a = torch.tensor([2.])
>>> b = torch.tensor([3.])
>>> c = a + b
>>> d = b - 2
>>> e = c * d
>>> print(c, d, e)
tensor([5.], grad_fn=<AddBackward0>)
tensor([1.], grad_fn=<SubBackward0>)
tensor([5.], grad_fn=<MulBackward0>)
```



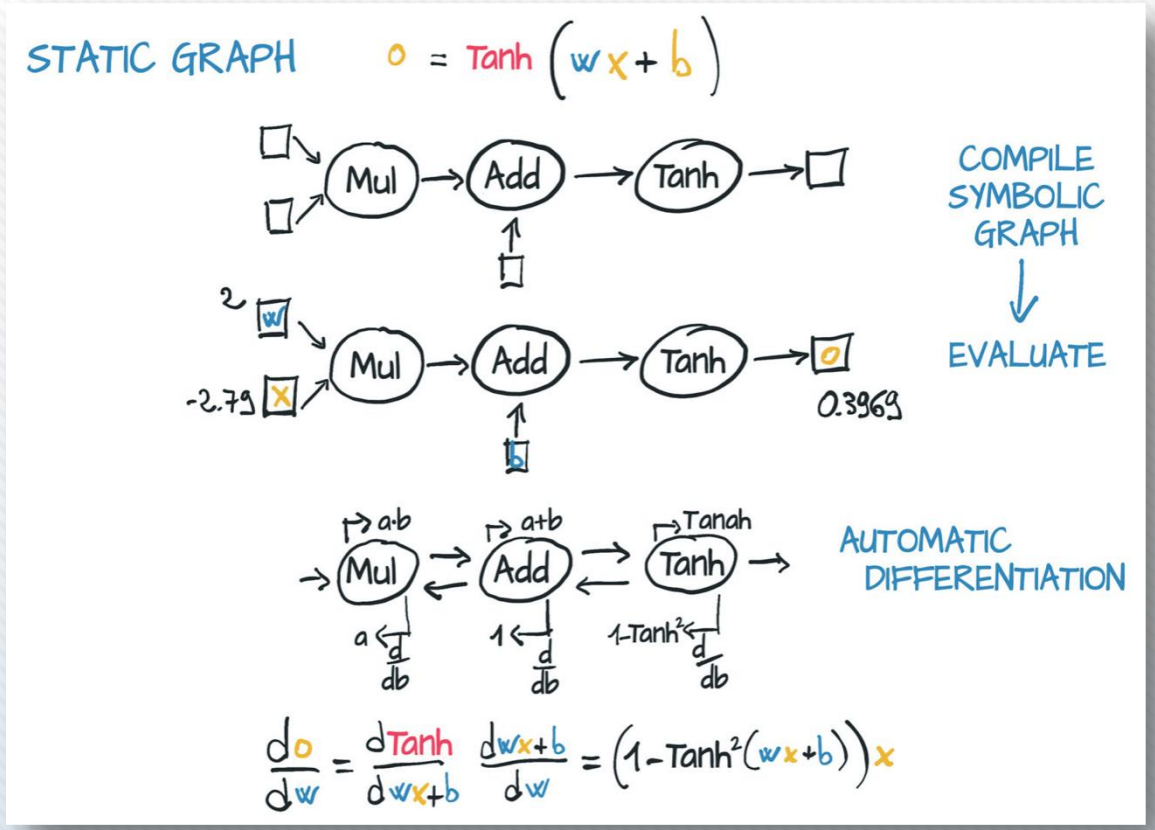
表达式（Expession）

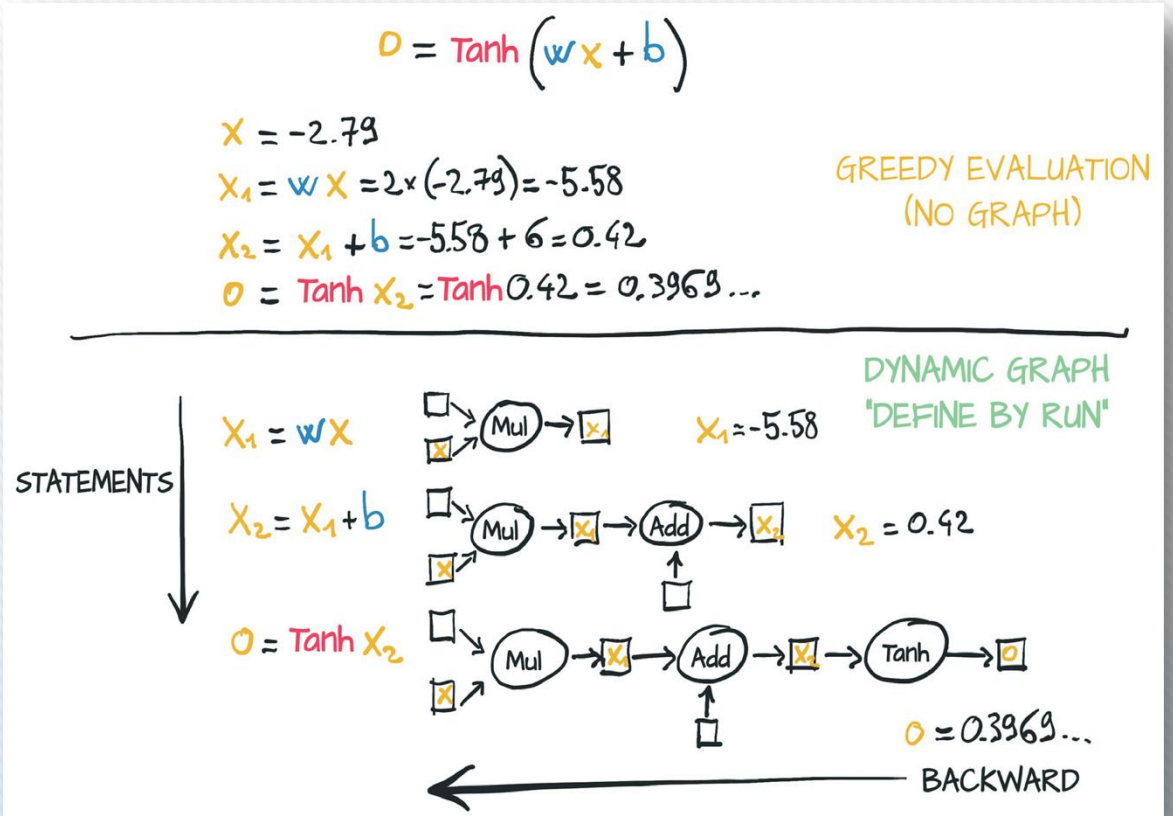
- 反向传播（Back-propagation）
  - 计算输出对每个输入的导数
    - $\frac{\partial e}{\partial a} = 1, \frac{\partial e}{\partial b} = 6$
- 自动微分计算

```
>>> e.backward()
>>> print(a.grad, b.grad)
tensor([1.]) tensor([6.])
```



静态图与动态图


STATIC GRAPH
o = Tanh (wx + b)
Mul
Add
Tanh
2
w
-2.79
x
b
0.3969
COMPILE SYMBOLIC GRAPH
EVALUATE
a·b
a+b
Tanah
a
1
1-Tanh²
AUTOMATIC DIFFERENTIATION
do/dw = dTanh/dwx+b · dwx+b/dw = (1-Tanh²(wx+b))x


o = Tanh (wx + b)
x = -2.79
x₁ = wx = 2×(-2.79) = -5.58
x₂ = x₁ + b = -5.58 + 6 = 0.42
o = Tanh x₂ = Tanh 0.42 = 0.3969...
GREEDY EVALUATION (NO GRAPH)
DYNAMIC GRAPH "DEFINE BY RUN"
STATEMENTS
x₁ = wx
x₂ = x₁ + b
o = Tanh x₂
x₁ = -5.58
x₂ = 0.42
o = 0.3969...
BACKWARD

模块（Module）

包名	功能	描述
torch	Tesnor / Expression	类似NumPy的张量库，支持GPU以及自动微分
torch.nn	Module	灵活的神经网络库，提供多种神经网络层
torch.optim	Module	多种优化算法
torch.utils	Module	数据集、数据加载等辅助工具

大规模预训练数据集

- 维基百科（Wikipedia）
  - 快照（2020年10月23日）https://dumps.wikimedia.org/

文件名	内容	大小/MB
zhwiki-latest-abstract.xml.gz	所有词条摘要	≈ 147
zhwiki-latest-all-titles.gz	所有词条标题	≈ 33
zhwiki-latest-page.sql.gz	所有词条标题及摘要	≈ 204
zhwiki-latest-pagelinks.sql.gz	所有词条外链	≈ 890
zhwiki-latest-pages-articles.xml.bz2	所有词条正文	≈ 1,952

  - 处理流程
    - 纯文本抽取 → 中文繁简转换 → 数据清洗
- Common Crawl数据（https://commoncrawl.org/）
  - 大规模网络爬虫数据集
  - 使用Facebook的CC-Net工具下载和处理
    - https://github.com/facebookresearch/cc_net

谢 谢!

理解语言，认知社会

以中文技术，助民族复兴

长按二维码，关注哈工大SCIR

微信号：HIT_SCIR