神经机器翻译实战 - 参加一次比赛

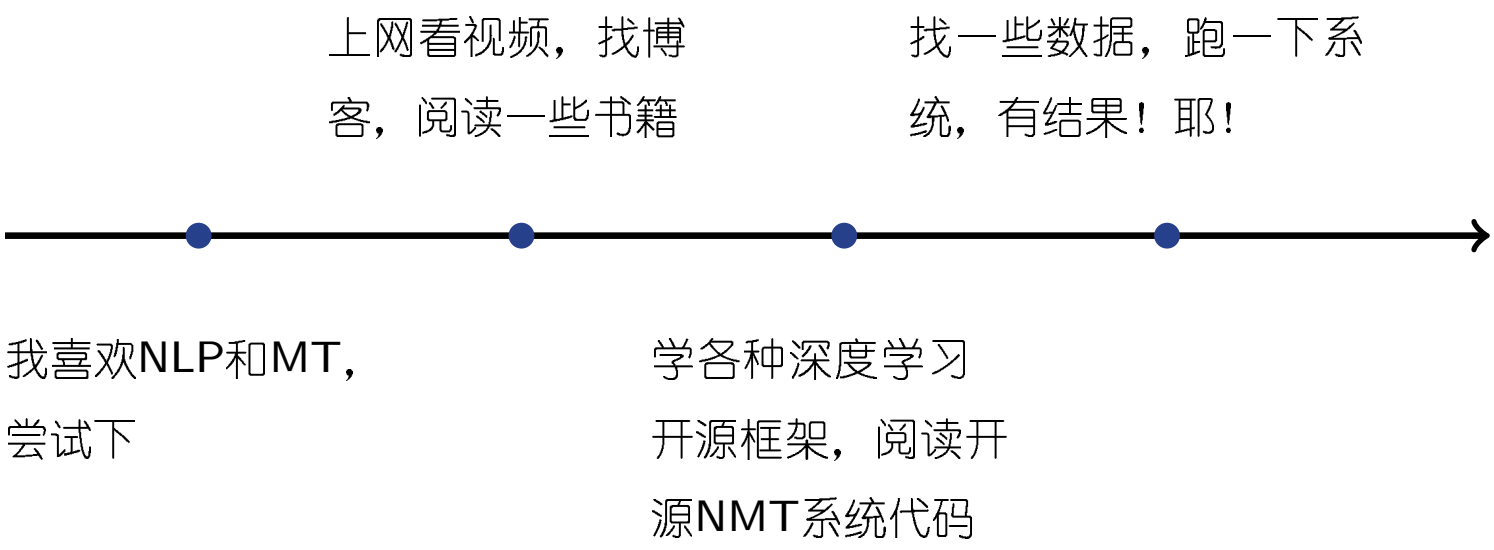
肖桐　朱靖波

xiaotong@mail.neu.edu.cn
zhujingbo@mail.neu.edu.cn

东北大学 自然语言处理实验室
http://www.nlplab.com

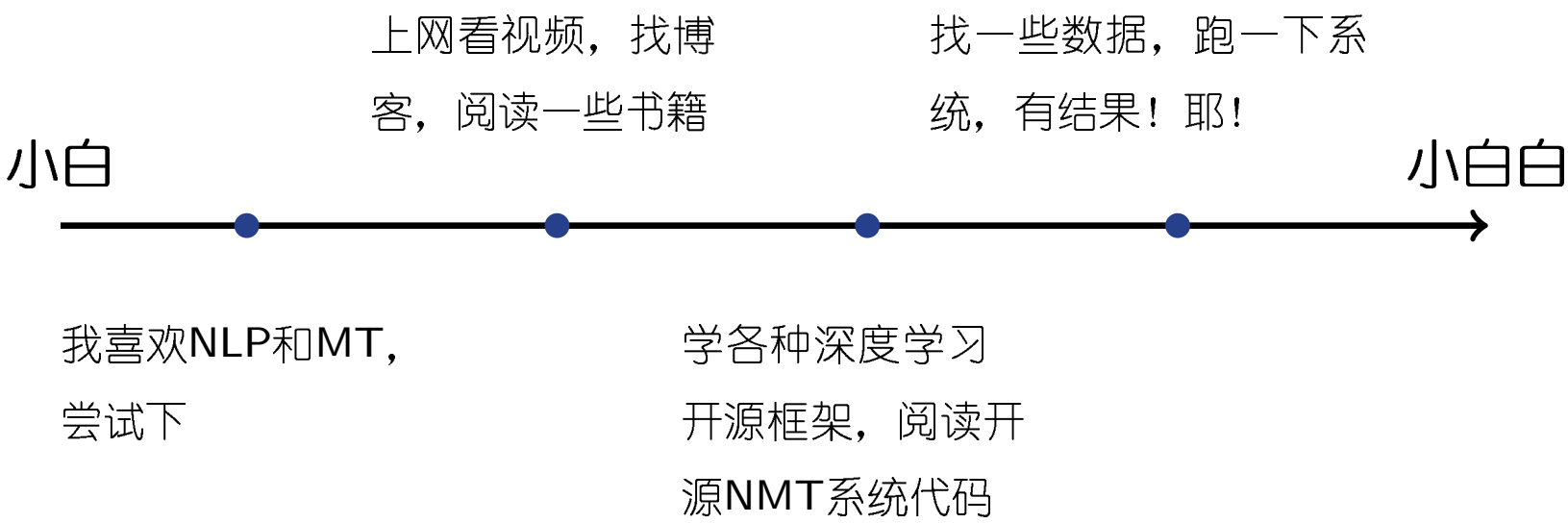
小白入门（神经）机器翻译

- 一个小白入门神经机器翻译的经历



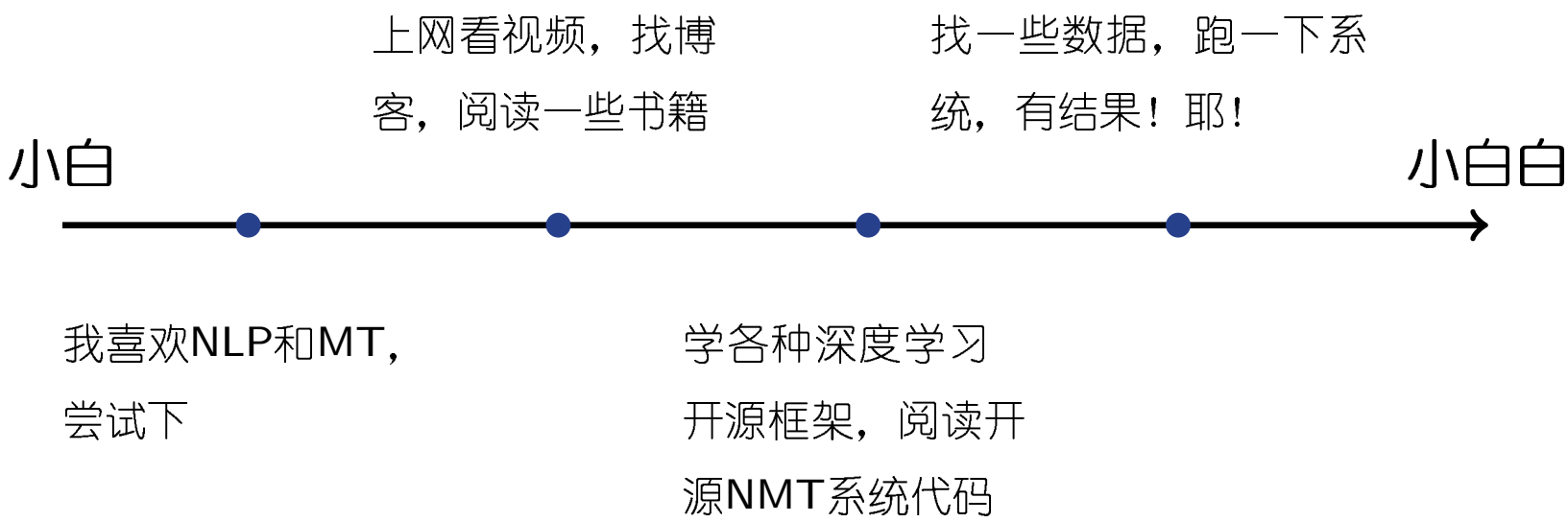
小白入门（神经）机器翻译

- 一个小白入门神经机器翻译的经历



小白入门（神经）机器翻译

- 一个小白入门神经机器翻译的经历



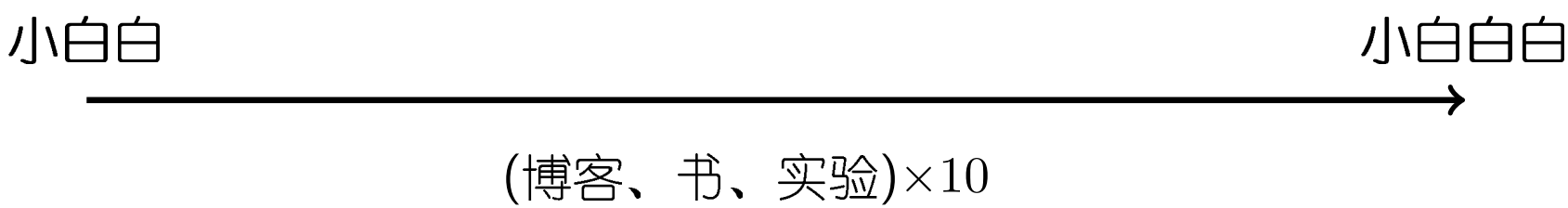
- 小白白还想成为大白，甚至大白白、大大白，还是使用同样的套路。

(博客、书、实验)×10

小白入门（神经）机器翻译

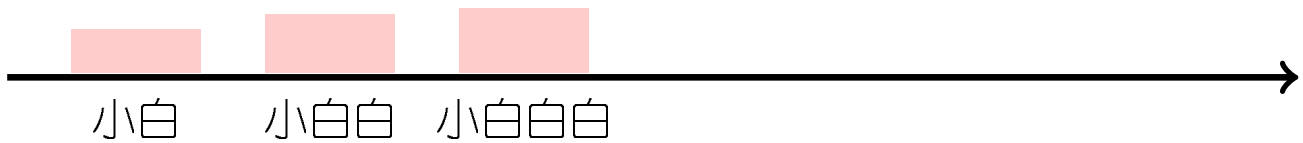
- 一个小白入门神经机器翻译的经历

- 小白白还想成为大白，甚至大白白、大大白，还是使用同样的套路。但是，只是变得更白了，离大白还很远!



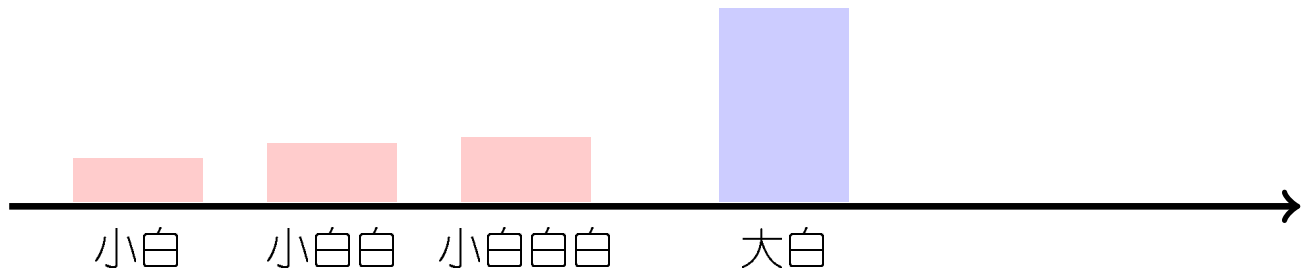
为什么小白没成为高手？

- 实力表



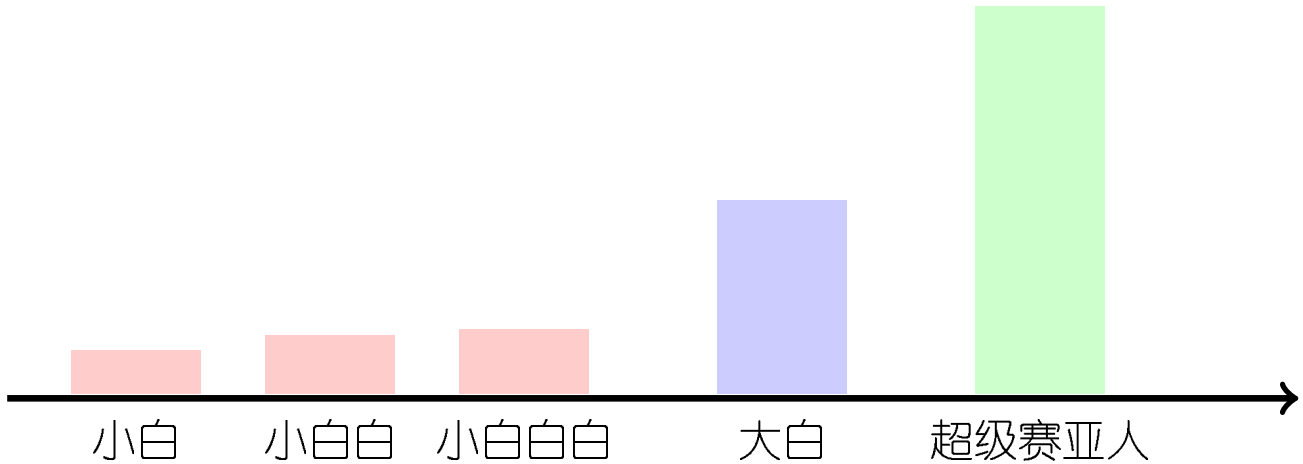
为什么小白没成为高手？

- 实力表



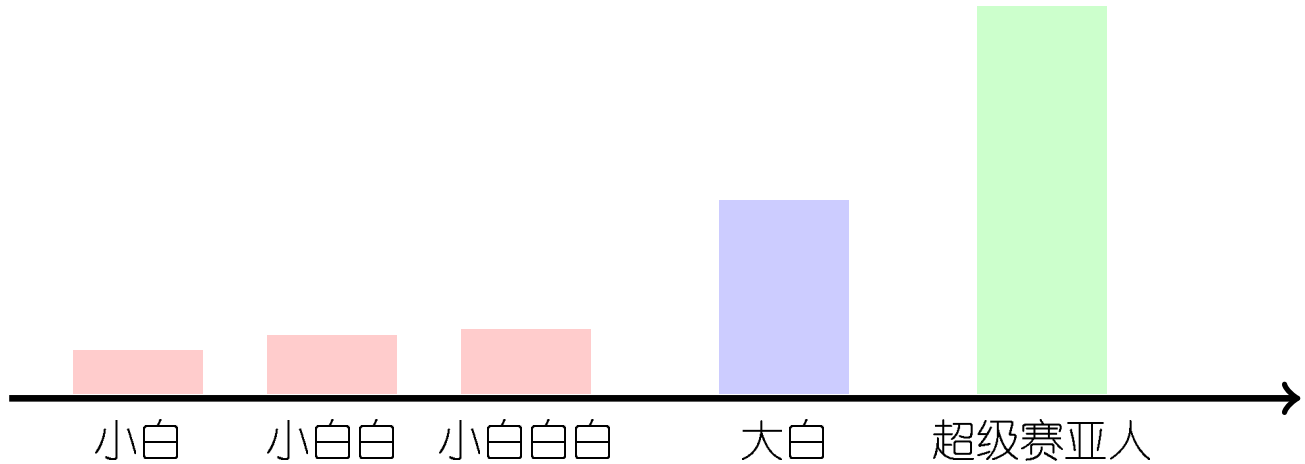
为什么小白没成为高手？

- 实力表



为什么小白没成为高手？

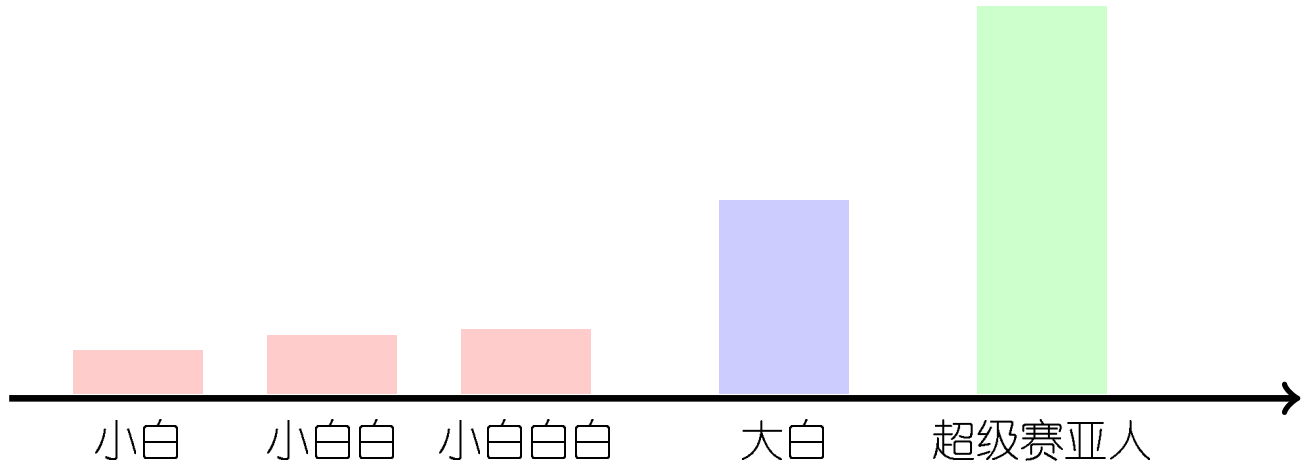
- 实力表



- 搞了半天，还是个入门选手，当遇到很多问题时还是无从下手
  - 论文写不出来，没有idea
  - 实验一弄就不好使，性能就是不涨
  - 自己搭的系统还很挫，离谷歌翻译还很遥远

为什么小白没成为高手？

- 实力表



- 搞了半天，还是个入门选手，当遇到很多问题时还是无从下手
  - 论文写不出来，没有idea
  - 实验一弄就不好使，性能就是不涨
  - 自己搭的系统还很挫，离谷歌翻译还很遥远
- 原因很简单：我们使用的系统还很初级，离state-of-the-art有距离，导致视野也会受限。
  - 怎么办？参加一次比赛吧

Towards State-of-the-Art

- 注意，机器翻译终究是面向应用的研究领域。因此，研发高性能的系统是我们追求的目标。一般来说，影响机器翻译系统性能有几个因素
  1. **数据**：大规模、高质量的数据是前提
  2. **技术**：算法和模型要足够先进
  3. **打磨**：需要对各个模块进行细致的打磨，说白了，下功夫不到有数据和技术也是白搭

Towards State-of-the-Art

- 注意，机器翻译终究是面向应用的研究领域。因此，研发高性能的系统是我们追求的目标。一般来说，影响机器翻译系统性能有几个因素
  1. **数据**：大规模、高质量的数据是前提
  2. **技术**：算法和模型要足够先进
  3. **打磨**：需要对各个模块进行细致的打磨，说白了，下功夫不到有数据和技术也是白搭

- 参加一次机器翻译的评测比赛是实现上述目标的一个很好的方法，比如，WMT News Translation Track
  1. 评测提供数据，规模和质量都能保证，因此可以在相对公平的数据基础上进行技术对比
  2. 评测系统都会有技术报告，因此可以相对容易的复现以前的结果，而且很容易了解最新的动态
  3. 评测本身就是驱动力（大家都希望取得好成绩），因此会不断打磨细节

基本流程

- 机器翻译比赛现在已经非常友好了（如WMT、CCMT等），有些比赛甚至都不需要提前报名，注册个账户就可以刷榜
  1. 从官方网站了解比赛计划
  2. 发放训练数据
  3. 数据初级及加工
  4. 选型，搭建第一个初级版本
  5. 和自己较劲（如果能够对比以前的参赛系统就更好了），改进系统，尝试各种技术
  6. 发放测试数据，运行系统提交最终结果
  7. 公布结果
  8. 撰写评测报告
  9. 参加评测研讨会，学习交流

基本流程

- 机器翻译比赛现在已经非常友好了（如WMT、CCMT等），有些比赛甚至都不需要提前报名，注册个账户就可以刷榜
  1. 从官方网站了解比赛计划
  2. 发放训练数据
  3. 数据初级及加工
  4. 选型，搭建第一个初级版本
  5. 和自己较劲（如果能够对比以前的参赛系统就更好了），改进系统，尝试各种技术
  6. 发放测试数据，运行系统提交最终结果
  7. 公布结果
  8. 撰写评测报告
  9. 参加评测研讨会，学习交流
- 当然，这里并不是要描述整个机器翻译的流程。我们重点还是关注有哪些技术可以使神经机器翻译变得更强，为相关研究提供合理的基线系统

必要的准备 - 选型

- 这里我们以WMT、CCMT的汉英新闻翻译任务为例，介绍如何搭建一套性能更加强劲的神经机器翻译系统
  - 翻译品质为评价指标（如BLEU），翻译速度等暂不考虑
  - 假设设备是充分的
  - 假设开发系统和模型训练时间也是充分的

必要的准备 - 选型

- 这里我们以WMT、CCMT的汉英新闻翻译任务为例，介绍如何搭建一套性能更加强劲的神经机器翻译系统
  - 翻译品质为评价指标（如BLEU），翻译速度等暂不考虑
  - 假设设备是充分的
  - 假设开发系统和模型训练时间也是充分的
- 面临的一个问题是选择什么样的系统架构。当然，这个问题可以边做表调整，不过这里我们选择Transformer作为基本框架（见第六章）。因为，
  - Transformer已经被证明是当今性能最好的NMT模型之一
  - 在WMT2019和CCMT2019的评测中，Transformer已经成为了各个参赛队伍的标配
  - Transformer是很多benchmark上的优胜者

必要的准备 - 选型

- 这里我们以WMT、CCMT的汉英新闻翻译任务为例，介绍如何搭建一套性能更加强劲的神经机器翻译系统
  - 翻译品质为评价指标（如BLEU），翻译速度等暂不考虑
  - 假设设备是充分的
  - 假设开发系统和模型训练时间也是充分的
- 面临的一个问题是选择什么样的系统架构。当然，这个问题可以边做表调整，不过这里我们选择Transformer作为基本框架（见第六章）。因为，
  - Transformer已经被证明是当今性能最好的NMT模型之一
  - 在WMT2019和CCMT2019的评测中，Transformer已经成为了各个参赛队伍的标配
  - Transformer是很多benchmark上的优胜者
- 当然，后面讨论的内容绝大多数与神经机器翻译架构无关，这些可以被推广到其它类型的系统上
  - **基础技术**：确保可以得到一个不太差的系统
  - **进阶技术**：接近甚至达到State-of-the-art

Outline

基础技术

1. 数据处理
2. 翻译单元切分
3. 正则化
4. 多模型集成
5. 自左向右解码 vs. 自右向左解码
6. 翻译长度控制
7. 大模型和大批量训练

进阶技术

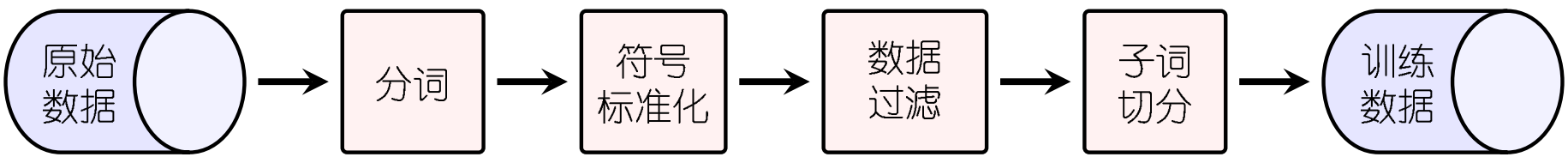
1. 深层模型
2. 单语数据的使用
3. 知识精炼
4. 双向训练
5. 统计机器翻译的使用

数据处理

- 双语数据的质量和数量对于训练一个神经机器翻译系统至关重要，有时即使拥有海量的数据，但是质量较差仍然会损失机器翻译的性能

		BLEU	
数据设置	数据量	英-德	德-英
初始数据	3M	32.5	31
+未处理额外数据	38M	26.6	32.7
+处理后数据	7M	37.9	37.5

- 数据处理便是对已有的数据集进行筛选处理的过程，一般的流程为：

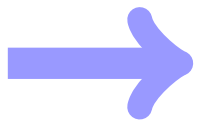


数据处理

- 神经机器翻译以词为单位处理句子序列，因此需要对以句子为单位的数据切分。
  - 以词表示的语言如英语，主要处理标点符号的粘连和短语的划分
  - 中文，日语等无明显词界限的语种，则需要特殊的处理。

中文：神经机器翻译改变了我们的生活。

英文：Neural machine translation has changed our lives.



中文：神经 机器 翻译 改变 了 我们 的 生活 。

英文：Neural machine translation has changed our lives .

- 常用的分词工具包括：mose，niutrans分词。。。

数据处理

- 符号标准化：统一源语目标语中的标点符号。
- 数据过滤是指过滤低质量数据,过滤的手段主要包括:
  - 重复数据过滤、乱码过滤、长度比过滤、HTML标签过滤、流畅度过滤。。。

源语	目标语
天气 好 但	The weather today is good , but . . .
我 喜欢 下雨 天 。	I like rainy days 。
<p> 显示 所选 的 面 。 <\p>	<p>to show the selected side . <\p>
桃树 、 杏树 、 梨树 ， 你 不让 我 ， 我 不让 你 ， 都 开满 了 花 。	Flowers bloom .
机器 翻译 给 人们 的 生活 带来了 便利 。	Machine translation brings convenience to people's lives.
这件 事情 的 成功率 为 50 o/o 。	The success rate for this matter is 50 o/o .
翻译 对 我 特别 感兴趣 。	I'm interested in translation .
他说 : “ 这个 深深 有趣 的 想法 印 在 我 心里 。 ”	He said: 、、This interesting idea is deeply imprinted in my heart. 、、
我 喜欢 下雨 天 。	I like rainy days 。
花下 成千成百 的 蜜蜂 嗡嗡 地 闹着 。	Hundreds of bees hummed under the flowers .

数据处理

- 符号标准化：统一源语目标语中的标点符号。
- 数据过滤是指过滤低质量数据,过滤的手段主要包括:
  - 重复数据过滤、乱码过滤、长度比过滤、HTML标签过滤、流畅度过滤。。。

源语	目标语
天气 好 但	The weather today is good , but …
我 喜欢 下雨 天 。	I like rainy days .
<p> 显示 所选 的 面 。 <\p>	<p>to show the selected side . <\p>
桃树 、 杏树 、 梨树 ， 你 不让 我 ， 我 不让 你 ， 都 开满 了 花 。	Flowers bloom .
机器 翻译 给 人们 的 生活 带来了 便利 。	Machine translation brings convenience to people's lives.
这件 事情 的 成功率 为 50 % 。	The success rate for this matter is % .
翻译 对 我 特别 感兴趣 。	I'm interested in translation .
他说 : “ 这个 深深 有趣 的 想法 印 在 我 心里 。 ”	He said: “ This interesting idea is deeply imprinted in my heart. ”
我 喜欢 下雨 天 。	I like rainy days .
花下 成千成百 的 蜜蜂 嗡嗡 地 闹着 。	Hundreds of bees hummed under the flowers .

数据处理

- 符号标准化：统一源语目标语中的标点符号。
- 数据过滤是指过滤低质量数据,过滤的手段主要包括:
  - 重复数据过滤、乱码过滤、长度比过滤、HTML标签过滤、流畅度过滤。。。

源语	目标语
天气 好 但	The weather today is good , but …
我 喜欢 下雨 天 。	I like rainy days .
<p> 显示 所选 的 面 。 <\p>	<p>to show the selected side . <\p>
桃树 、 杏树 、 梨树 ， 你 不让 我 ， 我 不让 你 ， 都 开满 了 花 。	Flowers bloom .
机器 翻译 给 人们 的 生活 带来了 便利 。	Machine translation brings convenience to people's lives.
这件 事情 的 成功率 为 50 % 。	The success rate for this matter is % .
翻译 对 我 特别 感兴趣 。	I'm interested in translation .
他说 ： “ 这个 深深 有趣 的 想法 印 在 我 心里 。 ”	He said: “ This interesting idea is deeply imprinted in my heart. ”
我 喜欢 下雨 天 。	I like rainy days .
花下 成千成百 的 蜜蜂 嗡嗡 地 闹着 。	Hundreds of bees hummed under the flowers .

翻译单元切分

- 对于某些形态学丰富语言，比如英语，德语等，常用一个单词的不同形态表示不同的意思，这就导致一个单词因为不同形态会产生各种不同词，就造成了大词汇量问题。

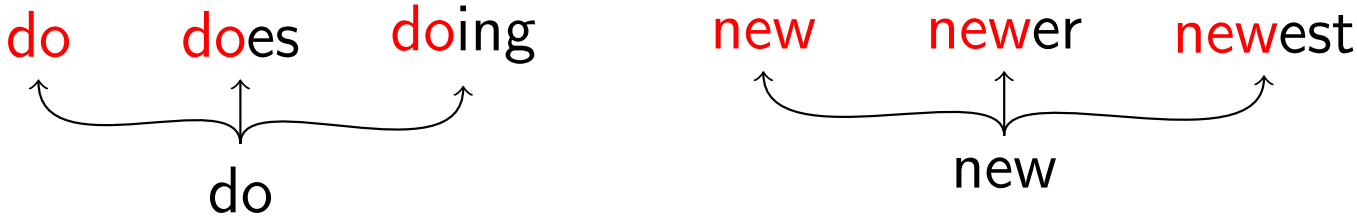
以英语为例：

名词	动词
cat，cats 、watch，watches	do，did ，does，doing，done
baby，babies、wife，wives	have，had，has，having

- 大词汇量导致翻译系统需要更大的词表，构造更大的词向量矩阵，带来了两个问题：
  - 词汇表稀疏，低频词得不到充分的训练。
  - 计算量更大，更大词向量矩阵，造成了计算资源的浪费。

翻译单元切分

- 为了解决这个问题，提出了子词的概念。



- 将词分解为子词的方式有很多种，主流的方法包括：
  - 字节对编码（ Byte Pair Encoding，BPE）– 统计频次
  - Wordpiece – 语言模型
  - Unigram Language Model – 语言模型
- BPE算法最早用于压缩，后来在NLP领域得到推广，是目前最流行，最简单有效的方法。
- 子词切分，缩减词表大小，节省计算资源，提高性能。
- 一般设定BPE切分后的词表大小为32000。

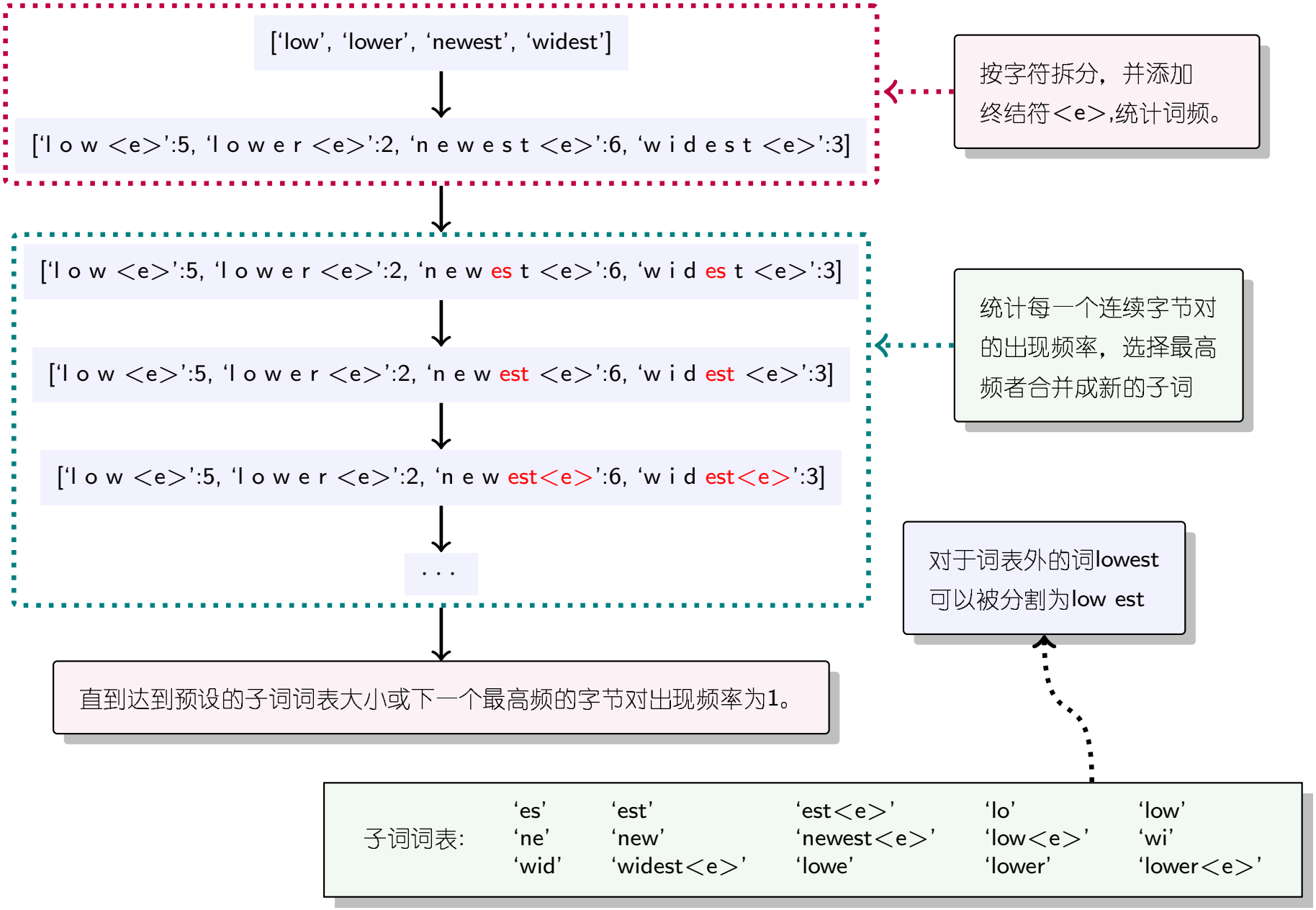
翻译单元切分

- 使用BPE切分的具体的流程如下：

- 在构建子词词表，应用BPE时存在两种方式：
  - 分别对源语和目标语构建子词词表，应用BPE。
  - 联合源语和目标语词表，称为Joint-BPE，可以增强源语和目标语的一致性。
- BPE能够有效减少词表大小，以及UNK的数量：

segmentation	# tokens	# types	# UNK
compound splitting	102 m	1 100 000	643
morfessor	109 m	544 000	237
hyphenation	186 m	404 000	230
BPE	112 m	63 000	0
BPE (joint)	111 m	82 000	32

翻译单元切分-子词构造


['low', 'lower', 'newest', 'widest']
['l o w <e>':5, 'l o w e r <e>':2, 'n e w e s t <e>':6, 'w i d e s t <e>':3]
按字符拆分，并添加终结符<e>,统计词频。
['l o w <e>':5, 'l o w e r <e>':2, 'n e w es t <e>':6, 'w i d es t <e>':3]
['l o w <e>':5, 'l o w e r <e>':2, 'n e w est <e>':6, 'w i d est <e>':3]
['l o w <e>':5, 'l o w e r <e>':2, 'n e w est<e>':6, 'w i d est<e>':3]
. . .
统计每一个连续字节对的出现频率，选择最高频者合并成新的子词
对于词表外的词lowest可以被分割为low est
直到达到预设的子词词表大小或下一个最高频的字节对出现频率为1。
子词词表:
'es' 'est' 'est<e>' 'lo' 'low'
'ne' 'new' 'newest<e>' 'low<e>' 'wi'
'wid' 'widest<e>' 'lowe' 'lower' 'lower<e>'

翻译单元切分

- 用构造的BPE词表，重新对句子中的词进行切分:
  - 对子词词表按长度由大到小排序

BPE词表:

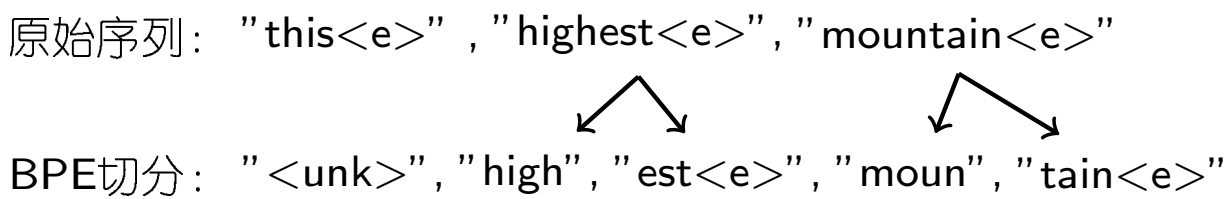
errrr<e>	tain<e>
moun	est<e>
high	the<e>
a<e>	

翻译单元切分

- 用构造的BPE词表，重新对句子中的词进行切分:
  - 对子词词表按长度由大到小排序
  - 遍历子词词表，寻找是否有当前单词的子串，如有则进行替换切分

BPE词表:

errrr<e>	tain<e>
moun	est<e>
high	the<e>
a<e>	

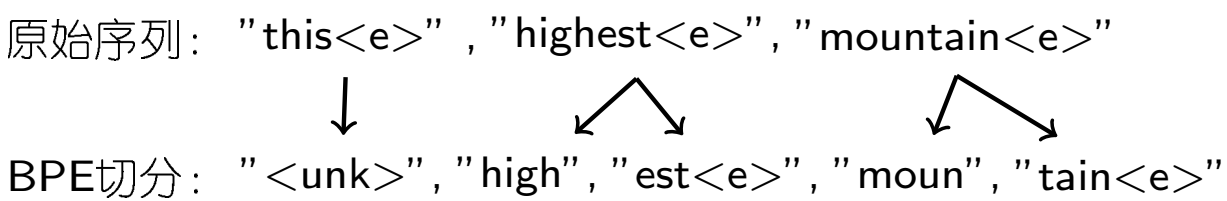


翻译单元切分

- 用构造的BPE词表，重新对句子中的词进行切分:
  - 对子词词表按长度由大到小排序
  - 遍历子词词表，寻找是否有当前单词的子串，如有则进行替换切分
  - 对子词词表中不存在的序列，替换为<unk>

BPE词表:

errrr<e>	tain<e>
moun	est<e>
high	the<e>
a<e>	

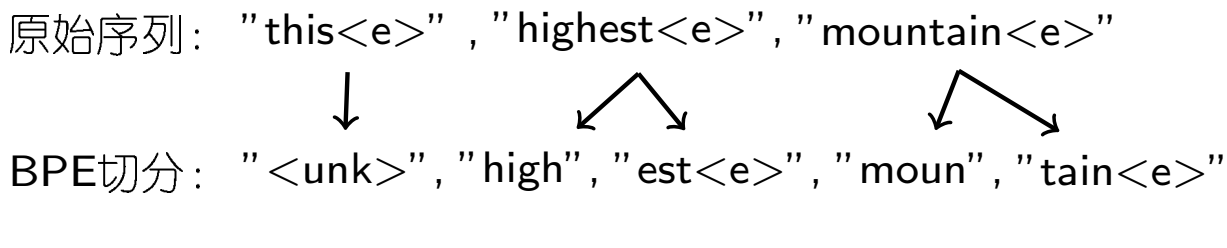


翻译单元切分

- 用构造的BPE词表，重新对句子中的词进行切分:
  - 对子词词表按长度由大到小排序
  - 遍历子词词表，寻找是否有当前单词的子串，如有则进行替换切分
  - 对子词词表中不存在的序列，替换为<unk>

BPE词表:

errrr<e>	tain<e>
moun	est<e>
high	the<e>
a<e>	

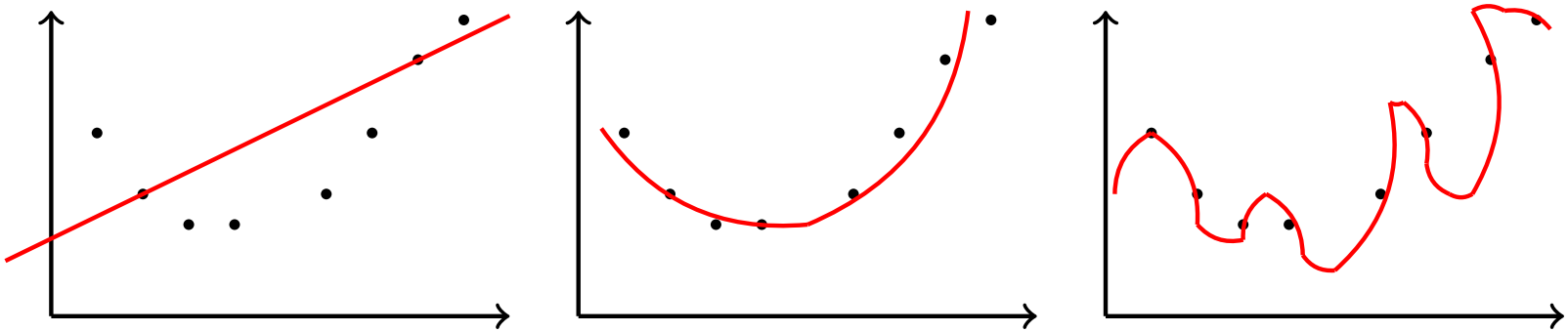


- 机器翻译解码的句子由子词组成，需要进行BPE还原,将每个子词向后合并，直至遇到终结符

翻译结果：”moun”,”tain<e>” ⟶ BPE还原：”mountain<e>”

正则化

- 神经机器翻译模型十分复杂，由于观测数据的不充分，及数据中存在的噪声，导致其求解十分不稳定，产生过拟合的现象。



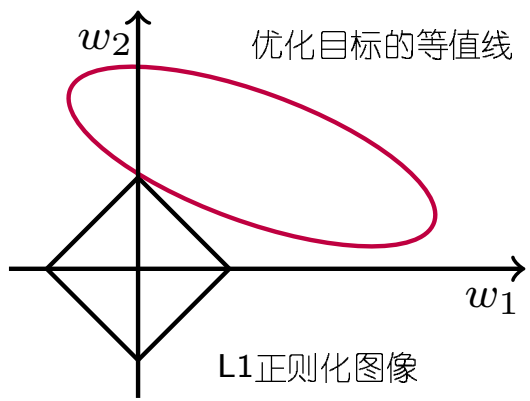
- 正则化是机器学习中的经典技术，通常用于缓解过拟合问题,即模型参数过度拟合噪声数据，因此正则化也常被称作降噪。
- 常用的正则化方法包括：调整训练目标、标签平滑、dropout。。。

正则化-优化训练目标

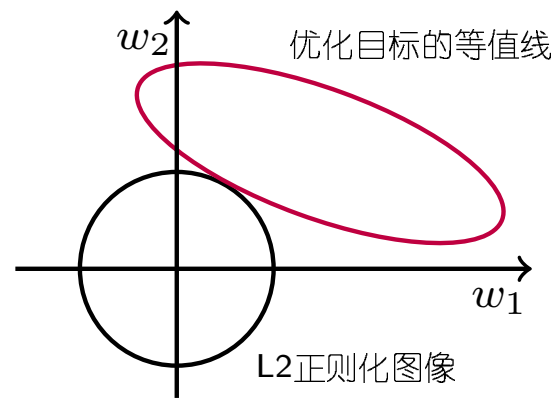
- 正则化的一种实现是在训练目标中引入一个正则项。在神经机器翻译中，引入正则项的训练目标为：

$$\widehat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} L(\mathbf{w}) + \lambda R(\mathbf{w})$$

- $L(\mathbf{w})$是损失函数，通过λ控制正则化项$R(\mathbf{w})$的强度
- 常用的正则化项包括：
  - L1正则化:$R(\mathbf{w}) = ||\mathbf{w}||_1 = \sum_{w_i} |\mathrm{w}_i|$
  - L2正则化:$R(\mathbf{w}) = (||\mathbf{w}||_2)^2 = \sum_{w_i} \mathrm{w}_i^2$



最优解处w1=0，L1正则化可以使参数矩阵更稀疏



切线处为最优解，L2正则化帮助缓解过拟合

正则化-优化训练目标

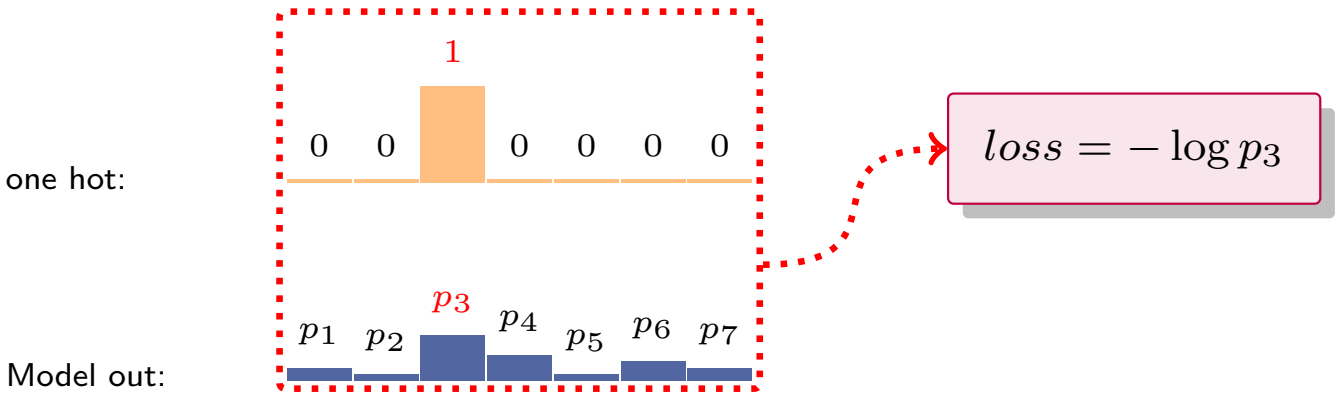
- 引入正则化项可以看作是使用了一种先验知识，使模型在求解时参数不会偏离0点太多，降低了模型的复杂度。

$$\underbrace{\begin{bmatrix} .1 & -4 & \cdots & 2 \\ 5 & 2 & \cdots & .2 \\ 2 & .1 & \cdots & .3 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & .8 & \cdots & 4 \\ -2 & .3 & \cdots & .1 \end{bmatrix}}_{\text{标准模型参数}} \qquad \underbrace{\begin{bmatrix} 0.11 & -0.4 & \cdots & 0.2 \\ 0.04 & 0.12 & \cdots & 0 \\ 0.23 & -0.1 & \cdots & 0.03 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0.28 & \cdots & -0.31 \\ -0.13 & 0.27 & \cdots & 0.01 \end{bmatrix}}_{\text{引入L2正则化}}$$

- 定义不同的正则化项，使模型偏向于我们希望的方向。
  - 通过约束层输出差异，增强模型的表现力
  - 修正注意力的分布，使其更关注全局或局部的信息
  - 增加层之间的辅助损失加快模型收敛。。。
- 除了将正则化项与损失线性结合的方式，我们还可以通过调整标准答案的方式来引入先验知识，如标签平滑

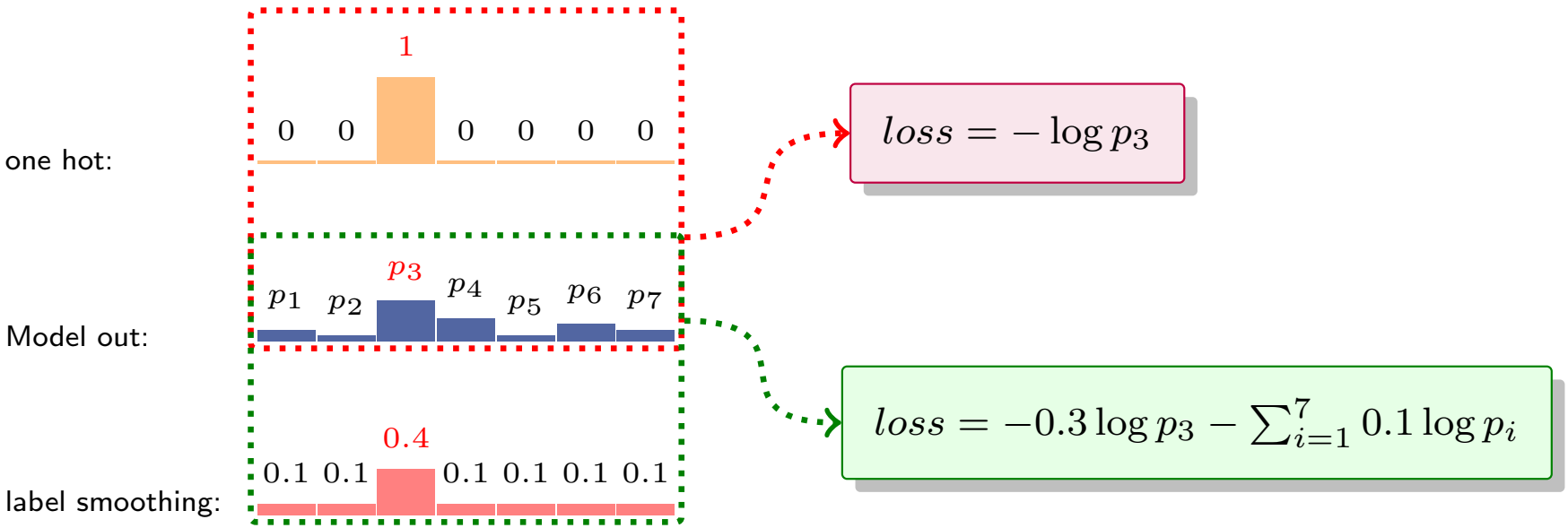
正则化-标签平滑

- 神经机器翻译系统对目标语的每个位置预测一个概率分布，表示词表中每个单词在该位置出现的可能性。
- 通过计算当前位置的分布与标准答案的差异作为损失
- 使用one-hot向量作为标准答案，不考虑类别间的相关性



正则化-标签平滑

- 神经机器翻译系统对目标语的每个位置预测一个概率分布，表示词表中每个单词在该位置出现的可能性。
- 通过计算当前位置的分布与标准答案的差异作为损失
- 使用one-hot向量作为标准答案，不考虑类别间的相关性
- label smoothing为所有位置分配了概率

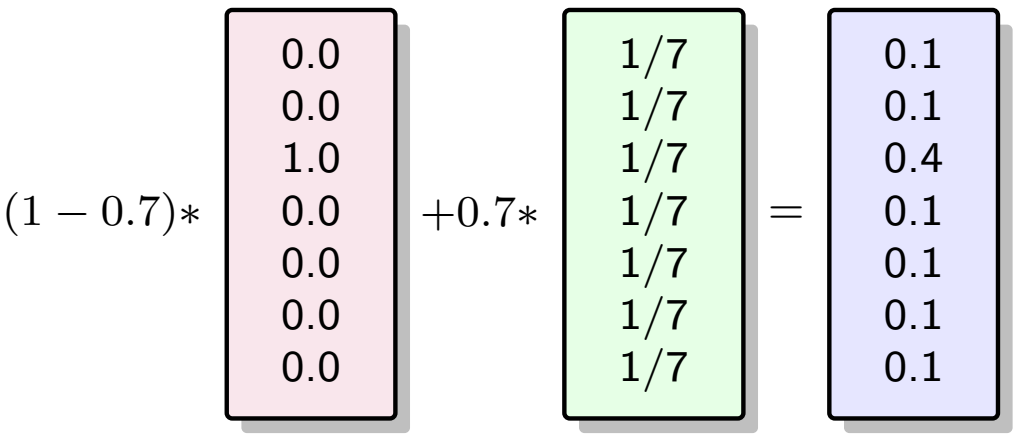


正则化-标签平滑

- 可以看到，label smoothing不会独享所有的概率。
- 在考虑正确答案的同时，也考虑了类别之间的相关性，提高了模型的泛化能力。
- 标签平滑的实现特别简单，只需要将真实答案独享的概率拿出一部分分享给所有的标签。

$$y_j^{\text{ls}} = (1-\alpha) \cdot \tilde{y}_j + \alpha \cdot q$$

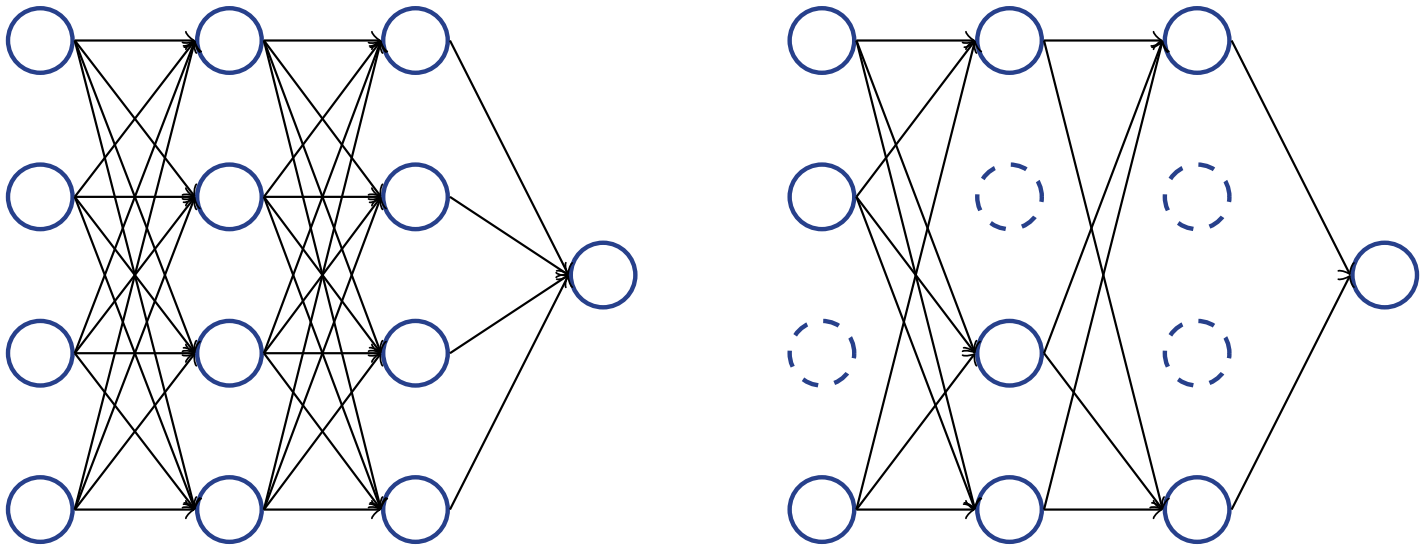
- 其中$\tilde{y}_j$为对j位置预测时的真实分布，即one-hot向量,q则是在词表大小V上的均匀分布



α表示分出去的概率大小
α越大，得到的分布越软
更软的分布更适合作为目标分布，可以缓解数据中噪声的影响

正则化-dropout

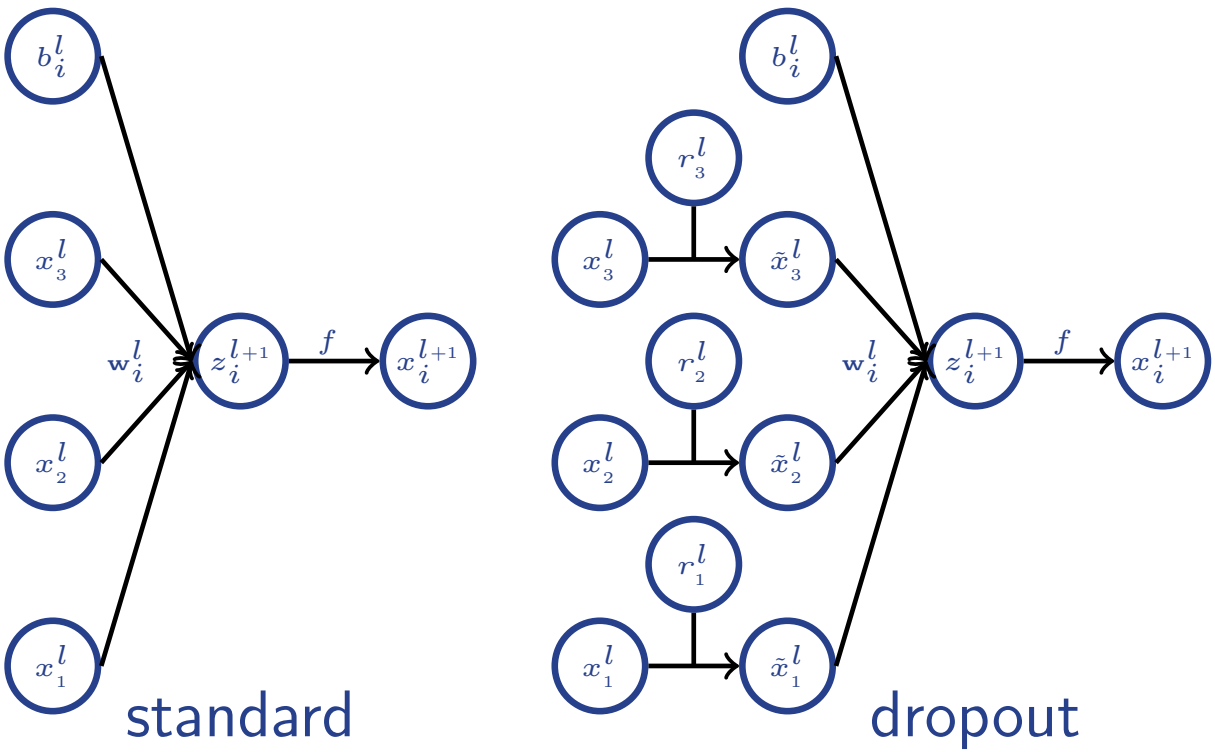
- Dropout通过在每一次的迭代中，只激活一部分节点，屏蔽其他神经元。相当于每次迭代训练的都是不一样的网络，从而降低节点之间的关联性以及模型的复杂度，达到正则化的效果。



- 简单的说就是，在前向传播过程中，以一定的概率p激活神经元，使其不过于依赖某些特征，增强模型的泛化能力。

正则化-dropout

- Dropout的工作流程如下：
  - 以一定的概率p随机的激活隐藏层神经元，掩盖其他位置神经元
  - 得到输入后进行训练，更新被激活的的神经元参数
  - 恢复被掩盖的神经元，此时，被激活的神经元已经更新
  - 不断重复此过程，直到训练结束



未应用dropout：

$$z_i^{l+1} = \mathbf{w}_i^l \mathbf{x} + b_i^l$$

$$x_i^{l+1} = f\left(x_i^l\right)$$

应用dropout：

$$r_j^l \sim \text{Bernoulli}\ (1-p)$$

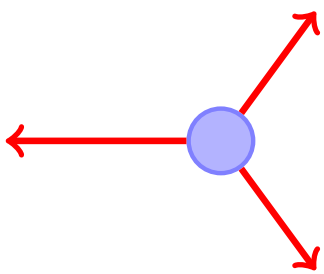
$$\tilde{\mathbf{x}} = \mathbf{r} * \mathbf{x}$$

$$z_i^{l+1} = \mathbf{w}_i^l \tilde{\mathbf{x}} + b_i^l$$

$$x_i^{l+1} = f\left(z_i^l\right)$$

正则化–dropout

- 在实现中，我们常常用p来表示神经元被掩盖的几率
- 训练阶段，通过dropout的方式类似于训练了不同的网络
- 推断时为了结果的稳定性，无法随机丢弃神经元，需要对此进行补偿:
  - 推断时对每个神经元乘以概率p
  - 训练时，对激活的神经元以$1/(1-p)$进行放缩
- 推断时，激活全部的神经元类似于对很不同结构的网络进行集成，使互为“反向”的状态抵消缓解过拟合现象。
- 根据数据量和模型结构合理的设置p能够有效缓解过拟合问题，提升神经机器翻译的性能。

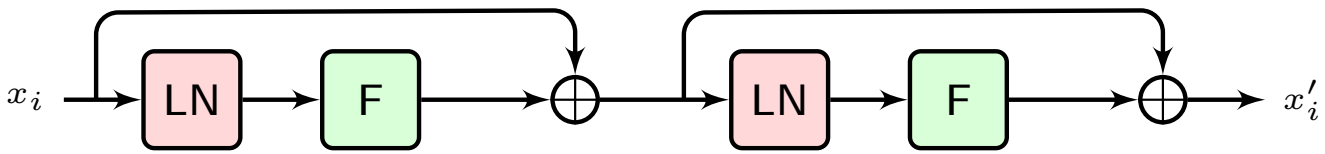


反向状态抵消

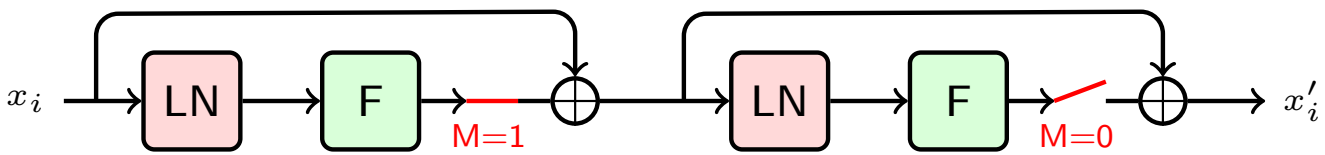
model	dropout	bleu
transformer-base	0	24.6
	0.1	25.8
	0.2	25.5
transformer-big	0.3	26.4

正则化-dropout

- dropout的思想同样可以应用在更高的维度，其中最常见的便是对模型结构的drop
- Transformer的结构由多层堆叠的编码解码层组成，残差连接组合各层的输出，不同层之间会相互影响
- 在深层Transformer结构中，更容易导致过拟合的现象
- 我们可以借鉴dropout的思想，对子层结构进行drop



Standtard Pre-Norm

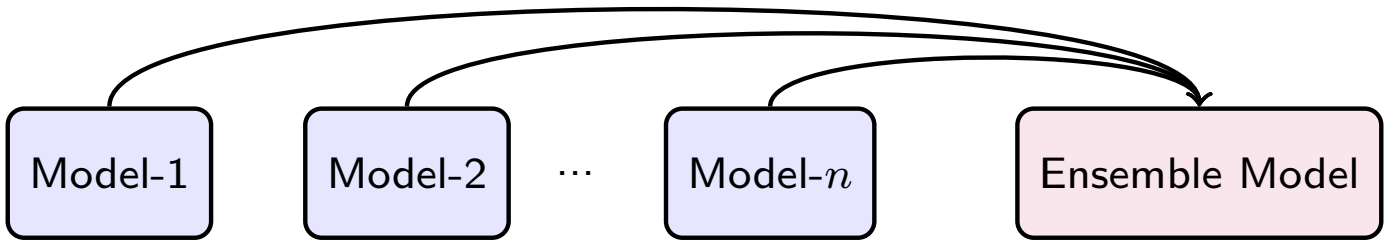


Pre-Norm with Skip sub-layer

- 对层结构的dropout，能够有效缓解深层网络中的过拟合现象，提升模型性能

多模型集成

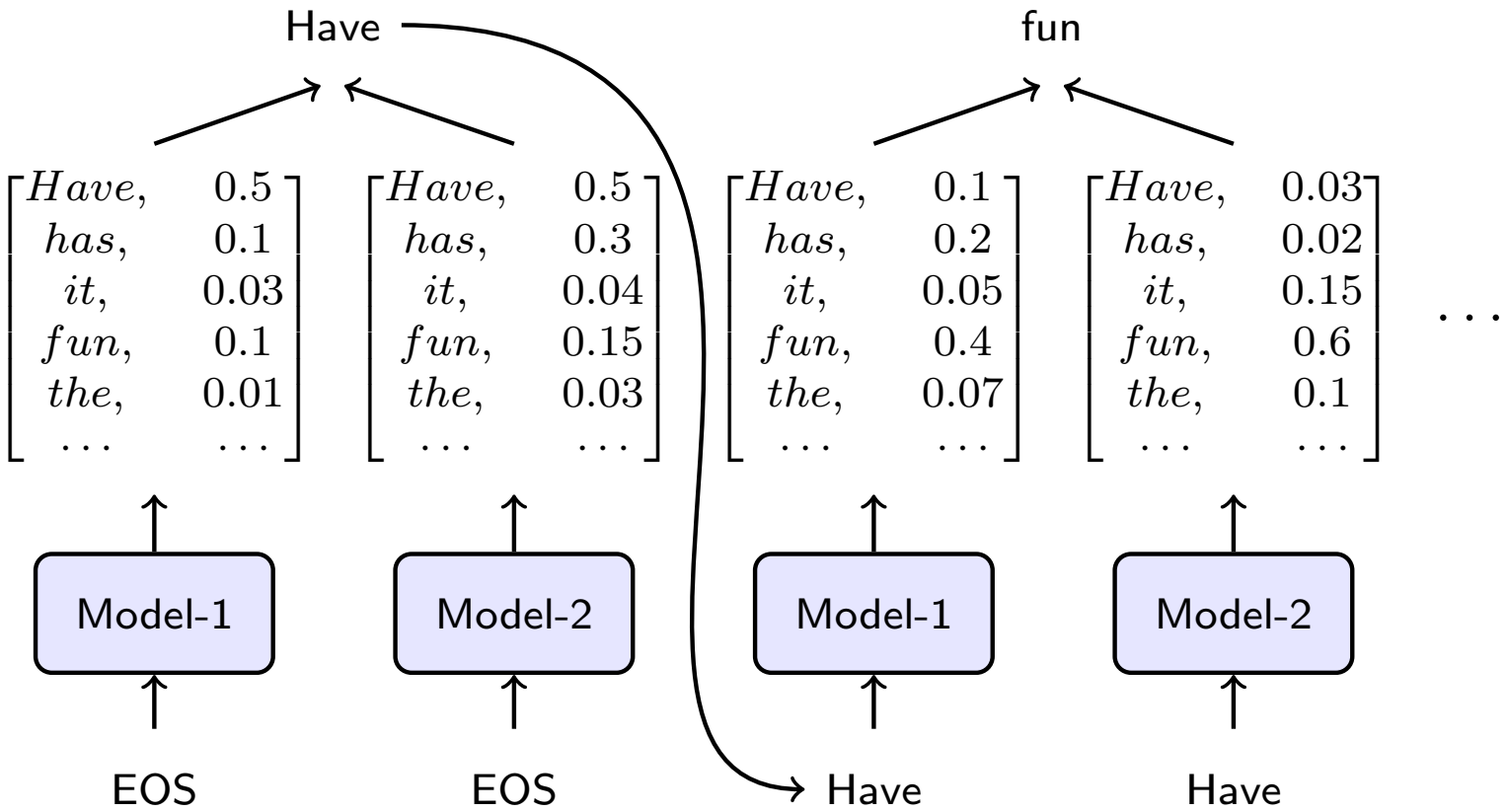
- 不同模型在训练的过程中的学习到的侧重面不一样。
  - 模型A：短语翻译的好
  - 模型B：单词翻译的好
  - 模型C：句子流畅度更好
- 模型集成便是通过对多个翻译模型的结果进行融合，各取所长，提升性能。



- 如何构造多个候选模型：
  - 通过不同的随机种子进行初始化，增加模型的多样性
  - 构造不同的模型结构，比如，相对位置建模，动态层聚合...
  - 使用不同的数据集

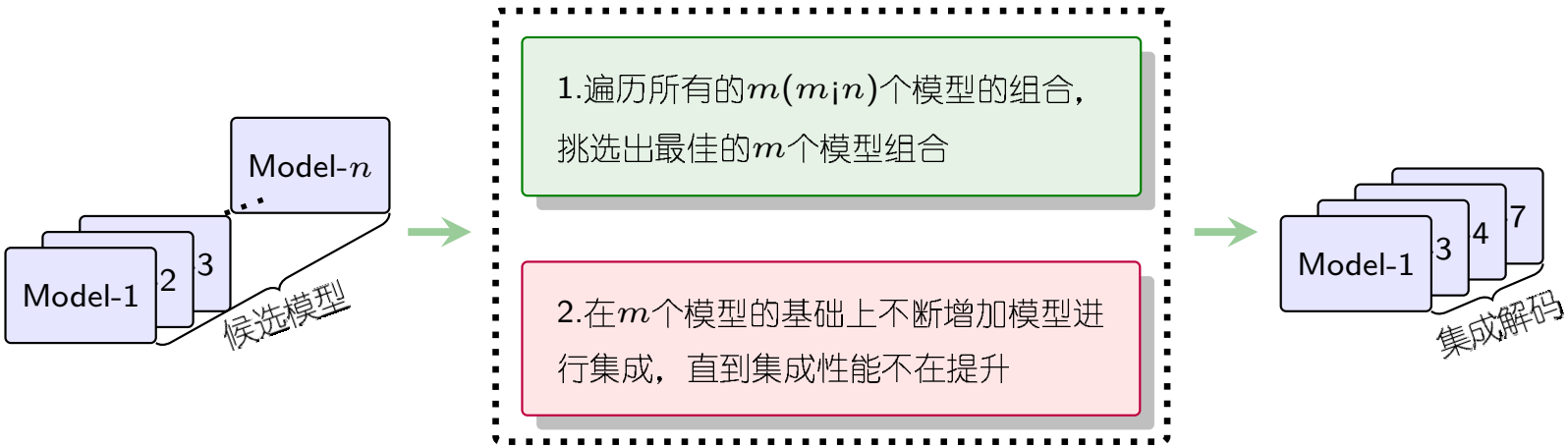
多模型集成

- 在机器翻译中如何实现多模型集成？
  - 检查点平均：对单个模型训练不同时刻保存的参数状态进行平均
  - 预测分布平均：对多个不同结构预测的概率分布进行平均



多模型集成

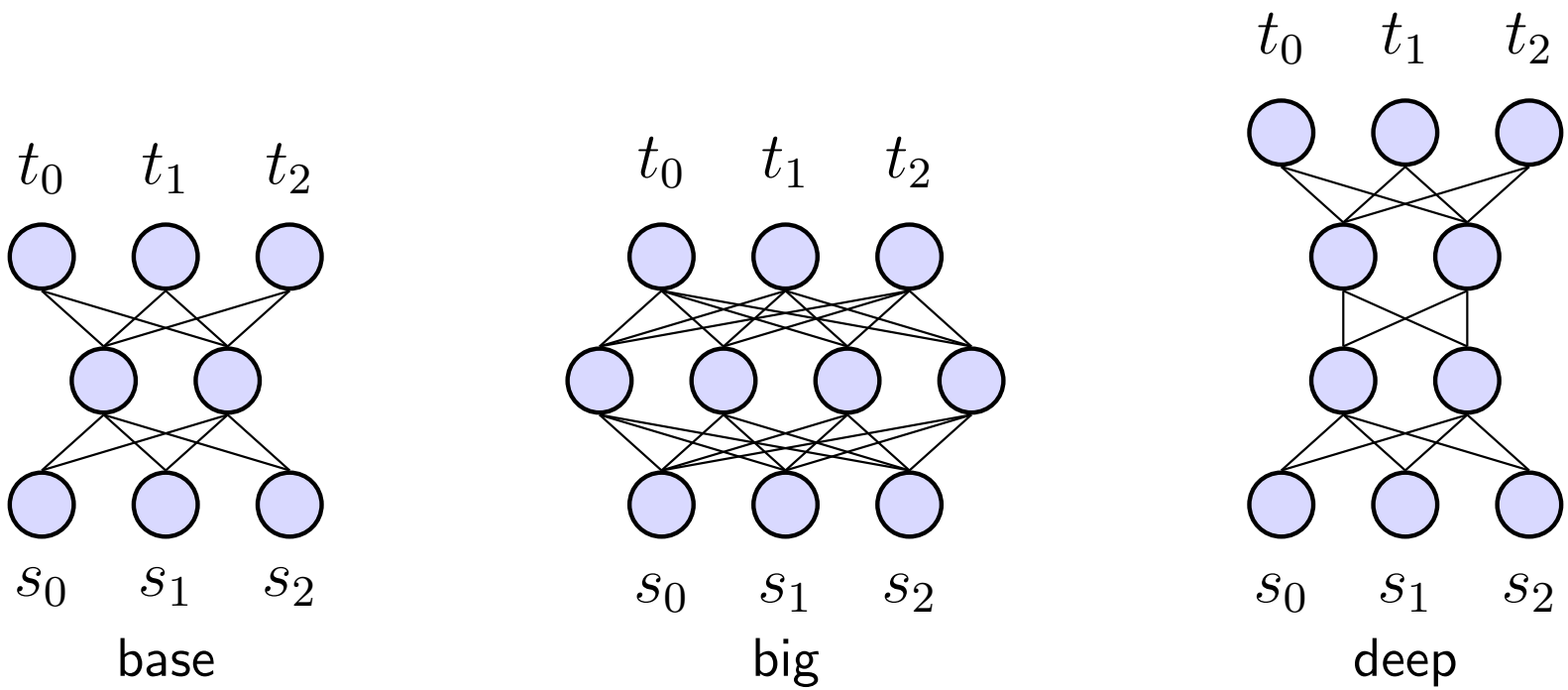
- 多模型的集成时，需要在众多的候选模型的选出最佳的集成组合。
  - 由于自回归特性，同时计算多个模型的概率分布，集成效率较慢，盲目尝试时间成本大。
  - 这里给出一种基于贪婪搜索的一种简单算法以便快速搜索出优秀的模型组合进行集成：



- 不是模型越多，集成性能越好，搜索范围可根据时间和计算成本确定

增大模型容量

- 模型容量与性能息息相关，在大规模的数据上训练时需要复杂的模型结构。
- 增大模型容量的方式主要包括：
  - 通过增大网络的隐层大小，即网络宽度
  - 增加网络的层数，即网络深度
  - 增大输入和输出层，即更大的词表和词向量维度



大模型-big model

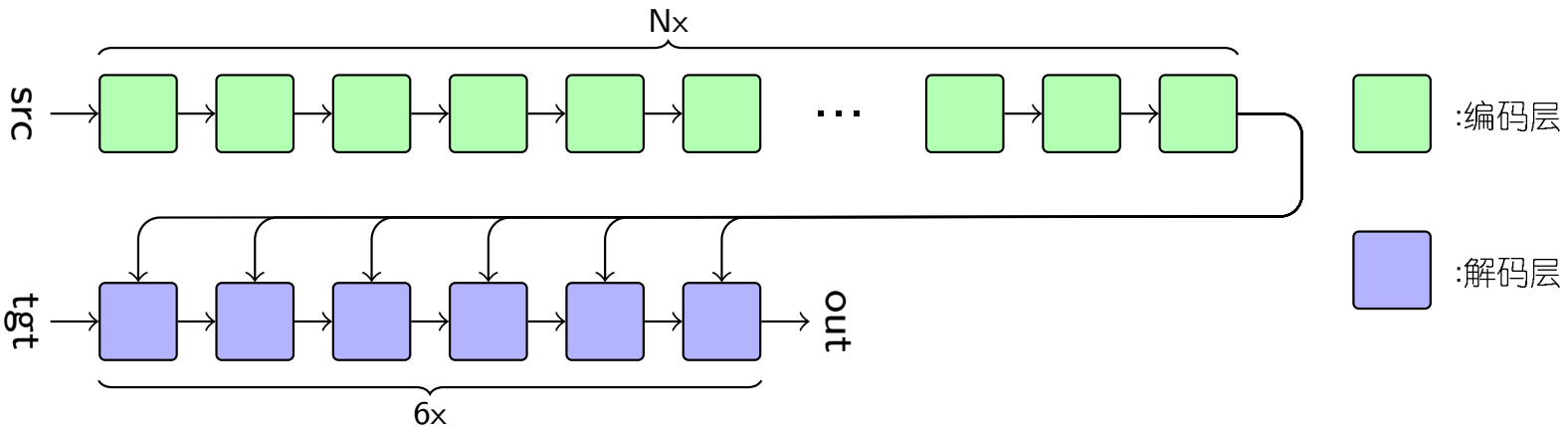
- 基于Transformer架构，我们增大其宽度的手段是使用Transformer-big模型。
- 通过增大模型的隐藏层大小，及输入输出层来提升模型容量，提升翻译品质。
- 除使用不同的参数设置外，transformer-base和big模型采用完全相同的网络结构。

	tranformer-base	transformer-big
词向量维度	512	1024
注意力头数	8	16
隐藏层维度	512	1024
FFN子层映射维度	2048	4096

- 采用transformer-big模型，同时针对不同的数据合理的调整学习率，dropout等参数，能够有效提升模型的性能。

大模型-deep model

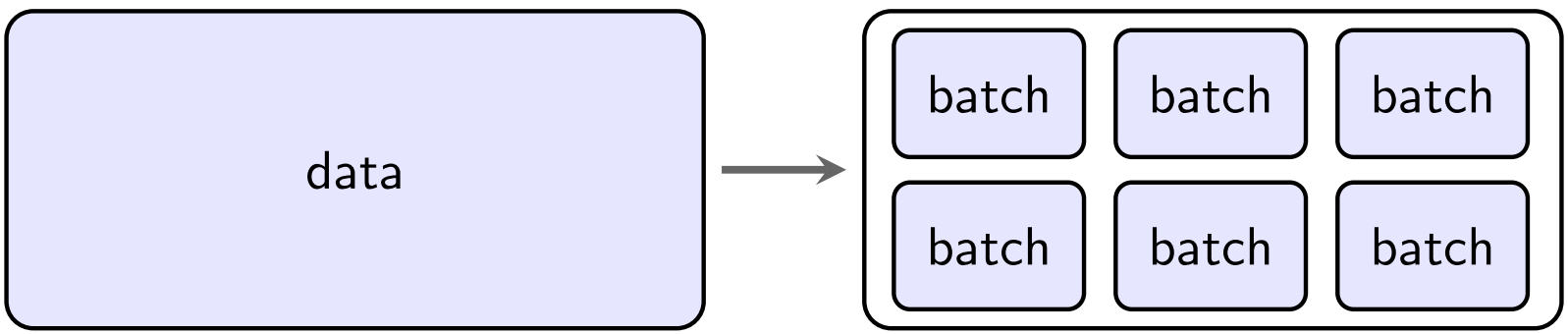
- 增大模型深度也是常用的提升模型容量的手段之一
- 对于transformer结构来说，增大模型深度指的是，增加编码端的编码层个数



- 增大transformer模型的编码层深度，通过更多的线形及非线形变换提升编码端的特称抽取能力。
- 单纯堆叠编码层，无法成功训练，通过合理的手段可以训练50层，甚至100层的模型。

大批量训练及解码

- 在人脑对文本进行处理时，通常以句子为单位，机器翻译中则通过批量的方式处理文本
- 批量的方法即使通过一定的方式将多个句子组合成一个批次送入模型
- 批量的方法可以用于模型的训练以及解码
  - 在训练时，同时输入多组源语和目标语，计算平均损失层，可以节省内存，加快收敛。
  - 解码时可以同时输入多句源语，得到一组译文，可以提高模型解码的效率



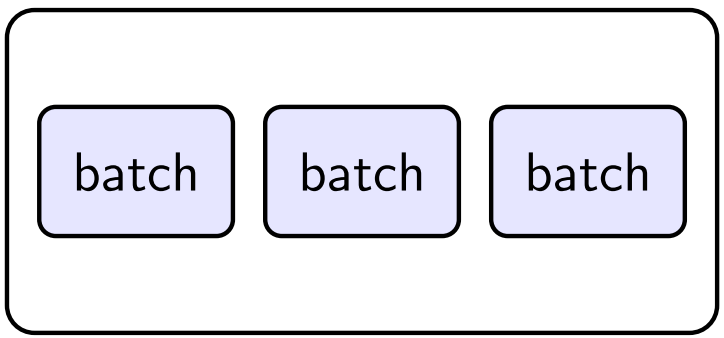
大批量-训练

- 逐句训练缓慢，在提出批量方法之前一次性把所有样本送入神经网络
  - 在整个语料库计算梯度，梯度方向更为准确
  - 大语料库导致内存爆炸，梯度间差异大难以使用全局学习率
- 如何合理构建batch十分重要，由于不同句子之间的长度有明显的的差异，使用padding对空白位置填充
- 由于padding机制，随机的生成batch会导致padding过多，收敛缓慢，对长度进行排序可有效缓解

- 除排序还有很多的策略用于batch生成，包括课程学习等，多种策略之间可以共同作用

大批量-训练

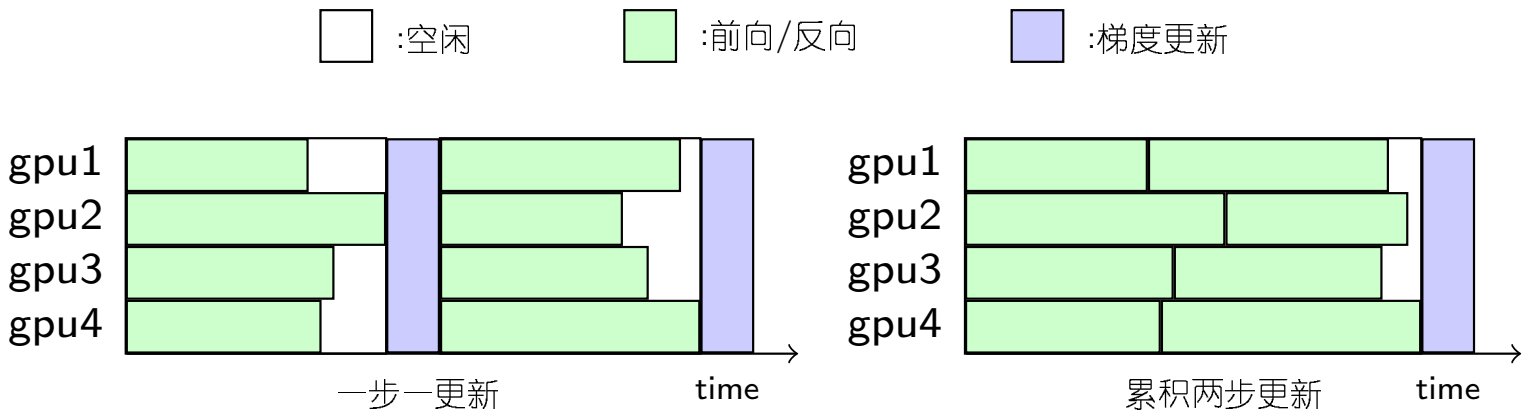
- 批量训练中一个重要的概念便是batch size，即每次送入模型的样本数量
- 除以句子作为度量单位外，常使用单词个数作为单位，batch size即为每次送入模型的单词个数
- 相比于句子的方式，进一步减少了padding数量，提高计算效率
- 实际中发现，训练时采用更大批量，配合更大的学习率，在加快模型收敛的同时有效提升了模型的性能。



batch	lr	BLEU
4096	0.01	29.15
8192	0.01	29.06
8192	0.02	29.49

大批量-训练

- 那么如何在有限的计算资源的情况下，增大batch size？
  - 多GPU以数据并行的方式分布式训练，同步更新，假设单块GPU容纳m个词，若拥有n块GPU，batch size便等于m*n
  - GPU的数量和内存受限时，可以采用累计梯度的方式
- 累计梯度是迭代多个批次，累积最终梯度进行更新
  - 累积n次，等价于batch size翻n倍
  - 减少设备间的通信，平衡多设备间运算差异，增大计算效率



大批量-解码

- 批量的方法同样可以应用于解码阶段，提升解码的速度和设备利用率
  - 批次生成的策略：常以句子数作为batch的度量单位
    - 当源语文本已知时，与训练类似，排序后划分batch
    - 实时翻译时，等待一个时间段，对期间得到的句子划分batch
  - 批次大小的设置：根据任务合理选择
    - 批量大，GPU利用率高，吞吐大，短句需要等待长句
    - 实时性要求高时不适合过大批量

实验分析

深层模型

反向翻译

知识精炼

双向训练

统计机器翻译的使用