AE语言模型和AR语言模型分析对比

一. AE和AR的不同点

1. **两类模型的建模方法不同**

- AR language model：生成式模型。对一个文本的概率分布进行建模，也就是将文本的概率从前向或者后向进行链式分解：

$$p(\mathbf{x}) = \prod_{t=T}^{1} p(x_t \mid \mathbf{x}_{>t}) \quad \text{or} \quad p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t \mid \mathbf{x}_{<t})$$

然后通过前向或者后向的文本表示来最大化下一个Token的log likelihood:

$$\max_{\theta} \quad \log p_\theta(\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(x_t \mid \mathbf{x}_{<t}) = \sum_{t=1}^{T} \log \frac{\exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x_t)\right)}{\sum_{x'} \exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x')\right)}$$

其中e代表单词的embedding，h代表前向文本或者后向文本模型编码后的隐藏层。

- AE language model：判别式模型。对一个有噪声的文本进行重建，本质上可以看做是一种去噪自编码器（DAE）：

$$\max_{\theta} \quad \log p_\theta(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_\theta(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x_t)\right)}{\sum_{x'} \exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x')\right)}$$

其中H代表模型对长度为t的文本的隐藏层编码表示，如果mt为1代表该位置的Token被MASK，AE LR通过上下文信息去除[MASK]噪声。

2. **两类模型获得的信息不同**

由于两类模型的建模方法不同，导致两类模型获得的上下文信息不同。

- AR language model：获得的是单向的上下文信息。
- AE language model：可以获得双向的上下文信息。

二. AR和AE的优缺点

- AR language model
  - 优点：由于采用单向概率建模的方式，对于自然语言生成类的下游任务效果良好。
  - 缺点：由于只能利用单向的上下文信息，导致全局信息利用率低，在一些自然语言理解类的下游任务表现较差。
- AE language model
  - 优点：可以充分挖掘文本的上下文信息，和AR模型相反，在自然语言理解类的下游任务表现好。
  - 缺点：首先是训练阶段假设单词之间具有独立性，但是这个假设太强，很多词之间具有关联性，其次预训练阶段的输入[MASK]在微调阶段不存在，这导致了预训练和微调两阶段之间存在差距。最后，在一些自然语言生成类的任务中由于不能利用文本的双向上下文信息，导致效果并不好。

三. AE模型和AR模型常见的架构和预训练任务设计

模型名称	语言模型	特征抽取器	预训练任务	主要特点
ELMo	AR	BiLSTM	Language Model	两个单向语言模型进行拼接
ULMFiT	AR	AWD-LSTM	Language Model	逐层解冻思想
SiATL	AR	LSTM	Language Model	逐层解冻+辅助LM
GPT	AR	Transformer Decoder	Language Model	首次引入Transformer Decoder
GPT2	AR	Transformer Decoder	Language Model	训练语料和模型深度的增加带来了性能的提升
BERT	AE	Transformer Encoder	MLM+NSP	引入MLM预训练任务和深度双向Transformer encoder
ENRIE1.0	AE	Transformer Encoder	BPE下的MLM	引入知识，从三个层次改进MLM任务
ENRIE2.0	AE	Transformer Encoder	MLM+多任务	引入多任务学习和连续增量学习
RoBERTa	AE	Transformer Encoder	Dynamic MLM+Model Input Format	精调BERT参数，优化BERT预训练任务
spanBERT	AE	Transformer Encoder	Span MLM	引入span masking和SBO损失，丢弃NSP
StructBERT	AE	Transformer Encoder	Word Structural Objective+ Sentence Structural Objective	引入句子中乱序单词和乱序句子两个预训练任务，适配下游任务
UniLM	混合	Transformer Encoder	Unidirectional LM+ Bidirectional LM+ Seq2Seq LM+span mask	将双向、单向、seq2seq三种训练方式通过注意力掩码结合起来，并且结合span Masking技术
BART	混合	Transformer	Token-masking+Token-Deletion+ Text-infilling+sentence-permutation+ Document-rotation	更多的预训练任务，对BERT结构进行进一步优化
MASS	混合	Transformer	Seq2seq LM	整合BERT和GPT的特点，融合了两者的优势
T5	混合	Transformer	Seq2seq LM	进行了充分的实验，实验基本覆盖预训练所有技巧
PEAGAUS	混合	Transformer	GSG+MLM	提出了GSG预训练任务，在摘要生成领域性能很好

从AE模型和AR模型的整体发展来看，AR模型早期主要是在模型结构上加深并且增加训练语料来达到模型的性能提升，使用的是Transformer Decoder作为特征提取，AE模型主要关注模型的预训练任务bert，通过引入一些预训练任务对一些特定的下游任务带来性能的提升，主要使用Transformer Encoder作为特征提取器，近期的预训练模型发展方向主要是两类模型的融合，例如UniLM，MASS等，采用的是Transformer encoder+decoder的结构，通过设计合理的预训练任务能够结合两种模型各自的优点。

由于AE模型和AR模型的本质差异，导致AR模型在自然语言生成类的下游任务如新闻、故事生成等方面效果比较好，AE模型则是在自然语言理解的下游任务如文本分类，问答，实体识别上效果好，另外的混合模型则是在一些seq2seq的任务如自动摘要生成，对话生成，机器翻译上表现良好。

还有一种特殊的排列语言模型（PLM）XLnet，使用双流注意力+部分预测+transformer-XL，由于排列语言的特性，融合了上下文的信息，由于采用transformer-XL，使得可以获得更长的依赖关系。后续的MP-Net提出了一种位置补偿策略，来弥补XLnet无法观测全局位置信息的问题。

从目前来看，AE和AR走向融合是一个必然的趋势，排列语言模型（PLM）是其中一种解决方案，但是在文本生成上没有充足的实验证明PLM性能的进步程度。

四. 一些预训练任务分类

预训练任务主要分成三种：

（1）Supervised learning

（2）Unsupervised learning（簇、密度、隐藏表示）

（3）Self-Supervised learning（MLM、NSP等）

下面是一些模型中用到的一些预训练方法：

1. Language Modeling (LM)

Probabilistic language modeling (LM)是最通用的无监督预训练任务，简单来说就是根据前n个单词或者后n个单词来预测当前位置上的Token.

2. **Masked Language Modeling (MLM)**

具体来说就是遮挡掉一部分token，然后试图使用其他的token预测遮盖掉的部分，这种预训练方法也有缺点，就是在训练阶段和fine-tune阶段的token出现数量不一致。

3. **Sequence-to-Sequence MLM (Seq2Seq MLM)**

S2S MLM通常被看做是一个分类任务，将一个masked的seq输入到encoder中，然后通过softmax输出分类概率，或者我们可以使用encoder-decoder架构，在encoder输入masked的seq，让decoder去生成对应的token（MASS、T5）。

4. **Enhanced Masked Language Modeling (E-MLM)**

对MLM任务进行提升，比如：

- RoBERTa使用动态mask；
- UniLM将MLM任务扩展为单向、双向、seq2seq三种；
- XLM使用双向平行语料库进行训练，XLM的每个训练样本包含含义相同语言不同的两条句子，而不是像BERT中一条样本仅来自同1一语言，XLM模型中，我们可以对每组句子，用一个语言的上下文信息去预测另一个语言被遮住的token。因为句子对中不同的随机词语会被遮住，模型可以利用翻译信息去预测token。模型也接受语言ID和不同语言token的顺序信息，也就是位置编码。这些新的元数据能帮模型学习到不同语言的token间关系。
- structBERT:1/3的时候： 是上下句，分类为1,1/3的时候： 是上下句反序，分类为2,1/3的时候： 是不同文档的句子，分类为3 这个任务对句子对相关的任务效果好。

5. **Permuted Language Modeling (PLM)**

为了解决在训练阶段的[MASK]模型在微调阶段不存在的影响，Permuted Language Model被提出来了，其中最有代表性的是XLnet.

6. **Denoising Autoencoder (DAE)**

输入一个带有噪声的seq，使用一个seq2seq模型去重建无噪声文本，有以下几种方法去破坏文本：

- Token Masking：随机采样token然后将它们替换成[mask]；
- Token Deletion：随机删除token，模型需要知道删除的token的位置；
- Text Infilling:具有代表性的是spanBERT，随机屏蔽一段text，模型需要预测多少text被屏蔽；
- Sentence Permutation:随机打乱句子的顺序；
- Document Rotation:随机选择一个token然后旋转documet，以这个token为开始，模型需要确定document的真实开始位置。

7. **Contrastive Learning (CTL)**

对比学习假设一些文本对相对于随机选择来说有更高的语义相似性。对于一个文本对，定义一个得分方程s(x,y)，任务的目标是最小化目标函数：

$$\mathcal{L}_{\text{CTL}} = \mathbb{E}_{x,y^+,y^-}\left[-\log \frac{\exp\left(s(x,y^+)\right)}{\exp\left(s(x,y^+)\right)+\exp\left(s(x,y^-)\right)}\right],$$

其中y+叫做和x相似的正样本，y-叫做和x不相似的负样本。得分方程通常通过learnable neural encoder学习，有两种方式，一种是分别将x和y进行编码后做点积：

$$S(x,y) = f_{\text{enc(x)}}^{T} f_{enc(y)}$$

另一种是向量拼接之后进行编码：

$S(x,y) = f_{\text{enc(x)}}^{T}(x \oplus y)$

CTL的想法是"learning by comparison"，通过对比学习，由于采用这种学习方式，CTL通常比LM的复杂度低。

8. **Deep InfoMax (DIM)**

最早是cv上面的概念，是通过最大化image representation 和local regions之间的互信息来提升image representation表达能力的。泛化到nlp里面就是把seq开头的字符（比如[cls]）的hidden state作为文本的编码表示，DIM的目标是最大化文本表示和噪声文本之间的互信息并且最小化噪声文本和随机采样文本之间的互信息：

$$\max f_{enc}\left(x_{i;j}\right)^{T} f_{enc}\left(\hat{x}_{i;j}\right) \; and \; \min f_{enc}\left(\tilde{x}_{i;j}\right)^{T} f_{enc}\left(\hat{x}_{i;j}\right)$$

其中$x_{i;j}$代表x文本i到j的n-gram span，$\hat{x}_{i;j}$代表MASK掉X中i到j的文本，$\tilde{x}_{i;j}$代表将i到j的文本替换成在语料库中随机采样一个负样本。

9. **Replaced Token Detection (RTD)**

主要思想和噪声对比估计（NCE）一样，但是预测的是给定一个token的上下文预测这个token是否被替换。CBOW可以看做是一个RTD的简易版本。

- ELECTRA：使用类似GAN的思想。
- WKLM则是在entity-level替换单词（替换的单词和被替换的词具有相同的类别，也就是相似度比较高），然后训练模型判断实体是否被替换。

10. **Next Sentence Prediction (NSP)**

50%替换成随意的下一个句子，但是后来的工作对于这个预训练任务基本上没有采用，效果也很好。

11. **Sentence Order Prediction (SOP)**

打乱句子的顺序进行训练，设计诸如预测句子是否被打乱等预训练任务。

从总体来说，模型的预训练任务十分庞杂，并且某个预训练任务不是”通吃“的，其在某个下游任务中表现良好，但是也有可能不适用于其他任务，所以结合预训练任务和下游任务也需要一定的经验。