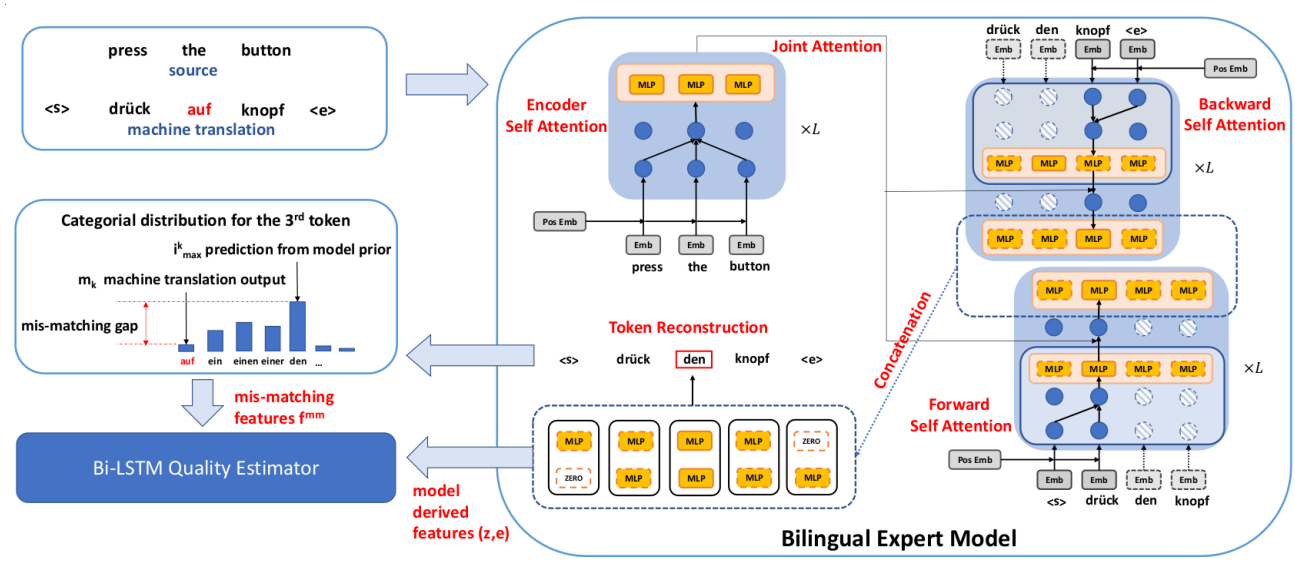
双语专家

翻译质量评估

- 句子级的评估: 回归模型
- 单词级的评估: 分类模型

系统



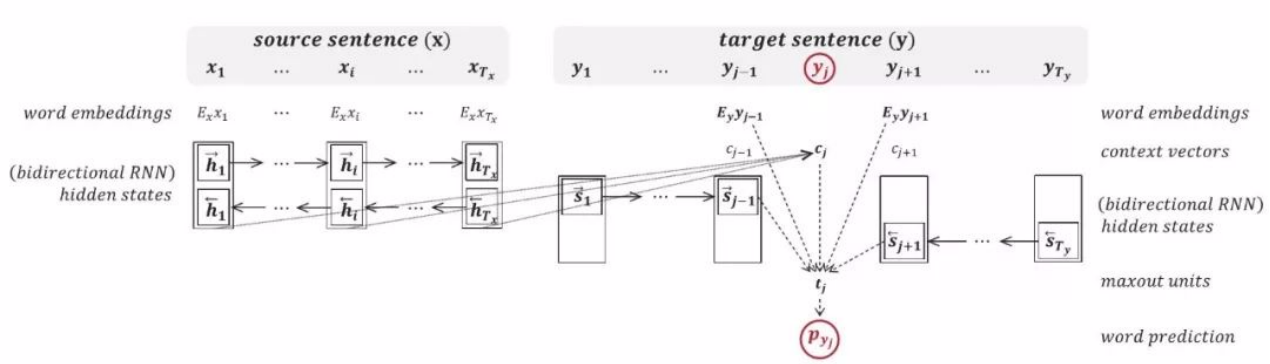
- 注意力机制加上Transfomer 机器翻译的模型框架，在前人研究的基础上提出了一种名为『Bilingual Expert』model (『双语专家』模型) 作为**特征抽取器**)，联合基于神经网络的**译文质量评估框架**。

特征抽取模型

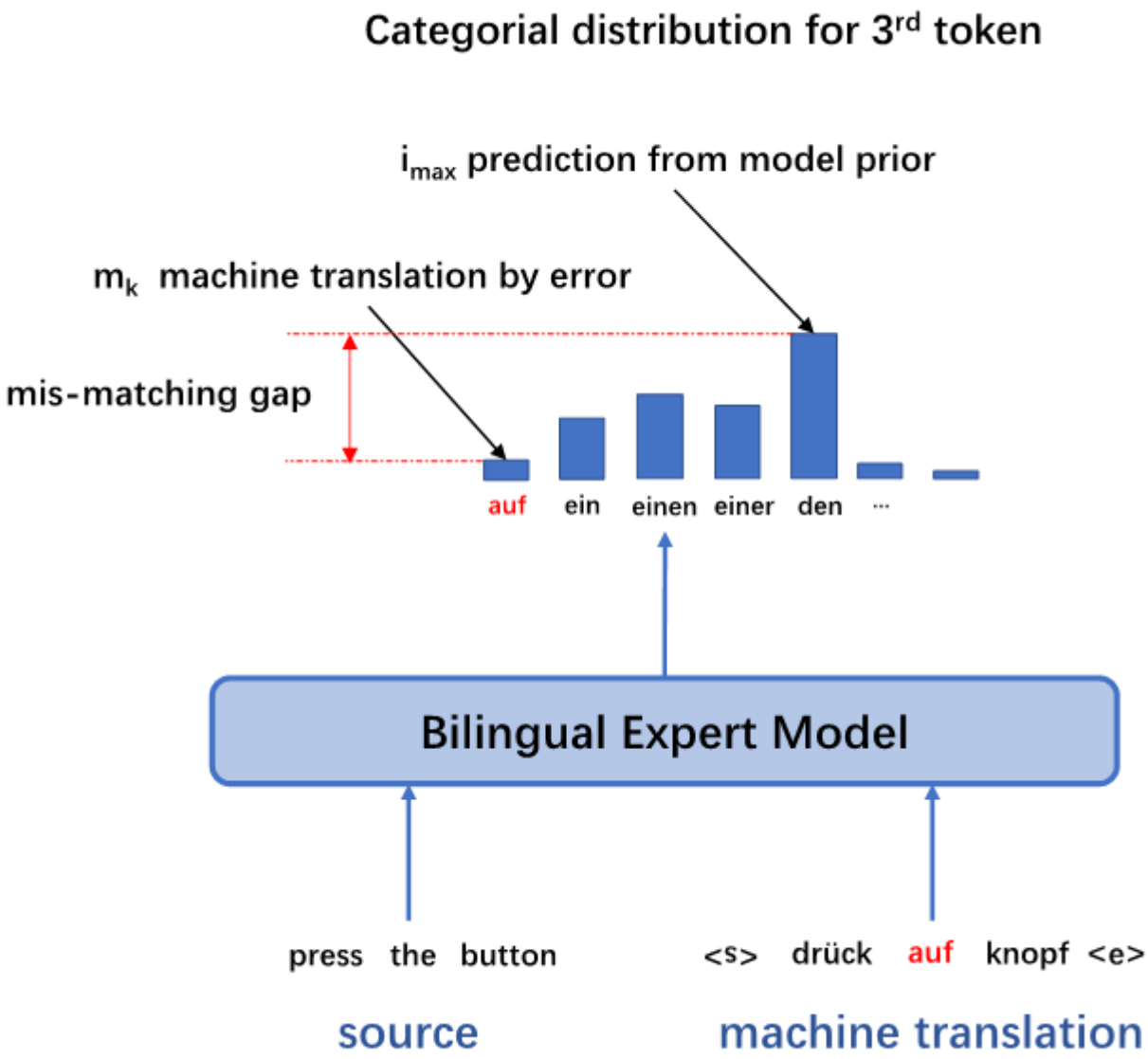
- 从原文与译文语句中抽取足够的信息或特征，并用来进一步计算译文效果到底好不好
- 特征抽取模型在输入原句序列和目标句序列的条件下抽取质量评估特征，这一部分的训练需要使用一般的双语平行数据集。而特征抽取模型抽取的特征可继续用于评估翻译效果，这一部分需要使用质量评估（QE）数据集，该数据集不仅包括原句与译文句，同时还包括了标注的翻译质量。
- 条件语言模型: 给定原语句子所有词和目标语句除当前词以外的上下文，模型希望能使用这些词的信息预测出当前词
- 如果译文的质量非常高，那么这种基于条件语言模型的词预测模型能基于原句子和目标句子的上下文准确预测出当前词。相反如果译文质量不高，那么模型很难基于上下文准确地预测出当前词
- 给定原语句子和目标语句子的上下文，并预测目标语句子的当前词可以表述为如下方程式

$$p(y|\mathbf{x}, \mathbf{y}_{-j}) = p(y|\, y_1, \cdots, y_{j-1}, y_{j+1}, \cdots, y_{T_\mathbf{y}}, \mathbf{x})$$

  - 传统的双向 LSTM 模型被替换为Transformer：挖掘序列内部的隐藏关系　+　提高并行效率，

- **传统LSTM的词预测模型(给定原语句子所有词和目标语句除当前词以外的上下文，模型希望能使用这些词的信息预测出当前词)**
- 

  - 对于原语句子 x，模型首先将每一个词都表征为词嵌入向量，然后再馈送到正向和反向两条 LSTM，每一个时间步需要同时结合正向和反向 LSTM 的隐藏状态并作为最终的输出。对于目标语句子 y，在第 j 个词之前的序列使用正向 LSTM 建模，而第 j 个词之后的序列使用反向的 LSTM 建模。最后**在预测第 j 个词时，需要使用原语句子 x 的上下文向量 c_j（由注意力机制得出）、目标语前一个词及前面序列的语义信息、目标语后一个词及后面序列的语义信息。**
- **阿里使用的Transformer架构：在原文和译文端之间进行注意力机制的计算，同时原文和译文内部也引入自注意力的机制，使得两端深层的语义信息能够很好得被学习到。**
- 编码过程：Transformer的编码过程，由相同的两个模块构成，每一个模块都有两个子层级。其中第一个子层级是 Multi-Head 自注意机制，第二个子层级采用了全连接网络，其主要作用在于注意子层级的特征。同时，每一个子层级都会添加残差连接和层级归一化。
- 解码过程: 创新之处，**基于 Multi-head Attention 的双向解码**，每个方向的解码器也由相同的两个模块堆叠而成，与编码器**区别**的是，每一个解码器模块都有**三个子层**组成。第一个和第三个子层分别与编码器的 Multi-Head 自注意力层和全连接层相同，而**第二个子层采用了 Multi-Head Attention 机制，使用编码器的输出作为 Key 和 Value，使用解码模块第一个子层的输出作为 Query**。与编码器**类似**的是，每一个子层同样会加上残差连接与层级归一化模块。该思想可以理解**构造了一个双向的 Transformer**，而其真正作用不是翻译系统中的解码器，而更像一个编码器或者特征表示器。
- **Transformer 编码器的 Inputs 为原语句子序列 x，解码器输入的 Outputs 为目标语正向和逆向两个序列**，解码器中 Softmax 输出的概率表示目标端当前词预测。在阿里采用的架构中，编码器和解码器的层数都等于 2
- **每一次在预测目标语的当前词时，Transformer 需要使用正向与反向两部分信息， 若当前预测目标语的第 j 个词，对于正向序列而言，模型需要使用目标端第 j-1 个词的正向深层语义特征向量$\overrightarrow{z_k}$和第 j-1 个词的词向量。而对于反向序列而言，模型需要使用目标端第 j+1 个词的反向深层语义特征向量$\overleftarrow{z_k}$与第 j+1 个词的词向量， 即token reconstruction**
- mis-matching features



- 当某个翻译结果错误单词不多的时候，预训练模型会给出正确的单词预测分布，这和翻译结果激活的单词会存在一个 gap。这个 gap 是一个非常重要的特征，阿里机器翻译团队的实验显示就算只用这个特征去做下一步预测，也可以得到很好的结果
- 在预测第 j 个词时，j+1 和 j-1 两个深层语义特征向量都相当于使用预训练的语言模型抽取语言特征，而那两个词的词嵌入向量则保留了原始信息。

- 除了需要预测最可能的当前词，更重要的是需要通过质量评估特征向量为后续运算迁移足够的语言知识。因此阿里的模型从词预测模型中抽取了两种质量评估特征

1. 深层语义特征

   - 正向深层语义特征向量 $\vec{Z}_k$
   - 反向深层语义特征向量 $\overleftarrow{Z}_k$

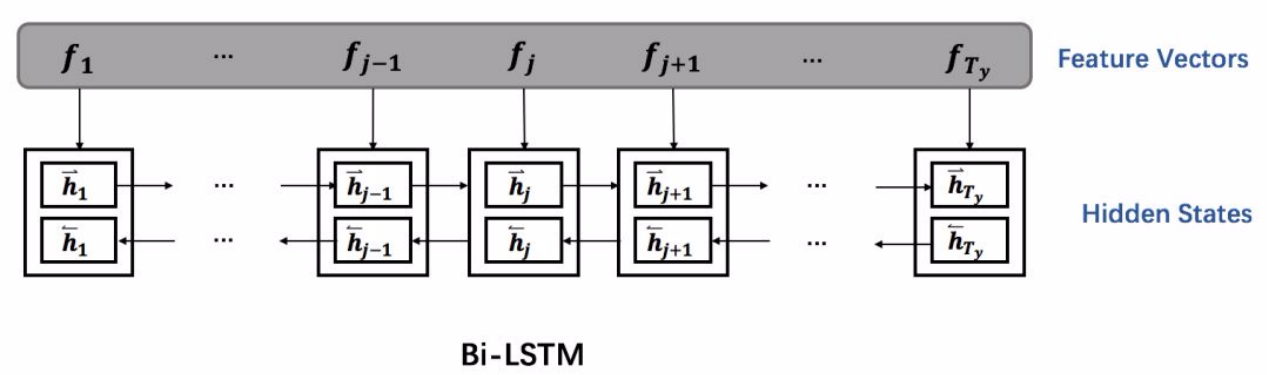
- 前一个词的词向量 $e_{t_{k-1}}$
- 后一个词的词向量 $e_{t_{k+1}}$

2. Mis-matching 特征
   - 目标端强制解码为当前词的概率信息 l_{k,m_k}
   - 概率最高词语的概率信息 $l_{k,i_{max}}$
   - 强制解码为当前词与解码为概率最高词的概率信息差异 $l_{k,m_k} - l_{k,i_{max}}$
   - 当前词与预测词是否一致 $\mathbb{I}_{m_k \neq i_{max}}$

- 其中正向和反向深层语义特征都从 Transformer 的解码器中抽出
  - 正向深层语义特征 $\vec{Z}_k$ **包含了原语序列的所有信息和目标语第 k 个词之前的语义信息**
  - 反向深层语义特征 $\overleftarrow{Z}_k$ **包含了原语序列的所有信息和目标语第 k 个词之后的语义信息**
  - 同时，深层语义特征还包含第 k-1 个词的词义信息 $e_{t_{k-1}}$ 和第 k+1 个词的词义信息 $e_{t_{k+1}}$
  - 在基于『双语专家』条件语言模型的词预测模型的预测解码环节，阿里机器翻译团队利用以上所有深层语义表达，重构了目标语 (Token Reconstruction)。所以如果我们强制解码为真实的词语，就可以取特征信息 l_{k,m_k}。不强制解码，保留模型预测最可能出现的词语，我们就能得到特征信息 $l_{k,i_{max}}$。剩下的两种特征则描述了 m_k 与 i_max 之间的关系。

质量评估模型

- 在抽取了质量评估特征后，它们可以与人工抽取的特征一起作为质量评估模型或 Quality Estimator 的输入来计算译文质量
- 条件语言模型的特征抽取模型和质量评估模型合在一起的缺陷：
  - 很多人工添加的特征是无法使用的(包括句长、标点符号数量、句子语言模型分数等17个特征)

    - 特征抽取模型广泛使用的平行语料与质量评估模型使用的 QE 数据集有比较大的不匹配性，联合训练可能会产生较差的性能。(平行语料只包含正确的目标语句子，而 QE 数据集同时包含正确与不正确的目标语句子)
- 将所有特征向量都拼接在一起，且每一个特征向量视为一个时间步，那么我们就能以如下方式利用从原文与译文中抽取的语义信息
    - 

    - 基于双向 LSTM，模型预测的目标即句子层面的翻译质量和单词层面的翻译对错, 这两个任务除了评估阶段采用的架构不一样，其它如特征抽取等过程都是一样的。
        - 在句子层面中，biLSTM 编码的前向的最后一个时间步与后向的最后一个时间步的隐藏特征联合计算一个实数值以表示翻译质量
        - 在词语层面, biLSTM 编码对应的目标端词的每一个时间步的前后向量隐藏特征联合计算一个值以将它们分类为 OK 或 BAD

数据集

- 词预测模型所使用的**平行数据集**和评估模型所使用的 **QE 数据集**
    - 平行数据集可以在广泛的领域收集, 目的是训练一个能抽取语言语义信息的模型，这很类似于预训练一个强大的语言模型。 QE 训练数据只有 1 至 3 万, 远远不够，通过**构造30万左右的QE训练伪数据**，这部分数据与真实 QE 数据合并训练完质量评估基线模型后，会再使用真实的 QE 数据微调模型，即使用一个在大的数据集上预训练好的模型在真实场景数据上微调。
- 训练伪数据：参考APE的方法，采用了一种 round-trip translation 的技术
    - 先从大量单语数据中筛选出领域相关的单语，作为人工后编辑译文 PE；同时用双语语料训练两个 MT 系统
    - 将筛选的领域单语先通过一个 MT 系统生成原文 SRC；SRC 再通过另一个 MT 系统生成译文 MT
    - 这样两次调取 MT 结果的方法，生成了一批原文，译文和人工后编辑译文组合的 APE 数据，称为 APE 训练伪数据， 然后通过 TER 工具生成了对应的 HTER 分数和词标注，构造出了 QE 伪数据
    - 为了更好地模拟真实数据，根据真实 QE 数据的 HTER 分布，从构造的伪数据中随机挑选出 30 万
    - 这些伪数据先与真实的 QE 数据一起训练一个 Quality Estimator 的基础 Baseline 模型，再单独用真实的 QE 数据 fine tune 模型

以下为论文

- **Kullback–Leibler divergence**：KL散度是两个概率分布P和Q差别的非对称性的度量。 KL散度是用来度量使用基于Q的编码来编码来自P的样本平均所需的额外的位元数。 典型情况下，P表示数据的真实分布，Q表示数据的理论分布。
    - 定义

- 对于离散型变量： $D_{\mathrm{KL}}(P\|Q) = -\sum_i P(i)\ln\frac{Q(i)}{P(i)}$. 等价于

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i)\ln\frac{P(i)}{Q(i)}.$$

- 对于连续随机变量，其概率分布P和Q可按积分方式定义为：

$$D_{\mathrm{KL}}(P\|Q) = \int_{-\infty}^{\infty} p(x)\ln\frac{p(x)}{q(x)}\,\mathrm{d}x,$$

其中p和q分别表示分布P和Q的密度。

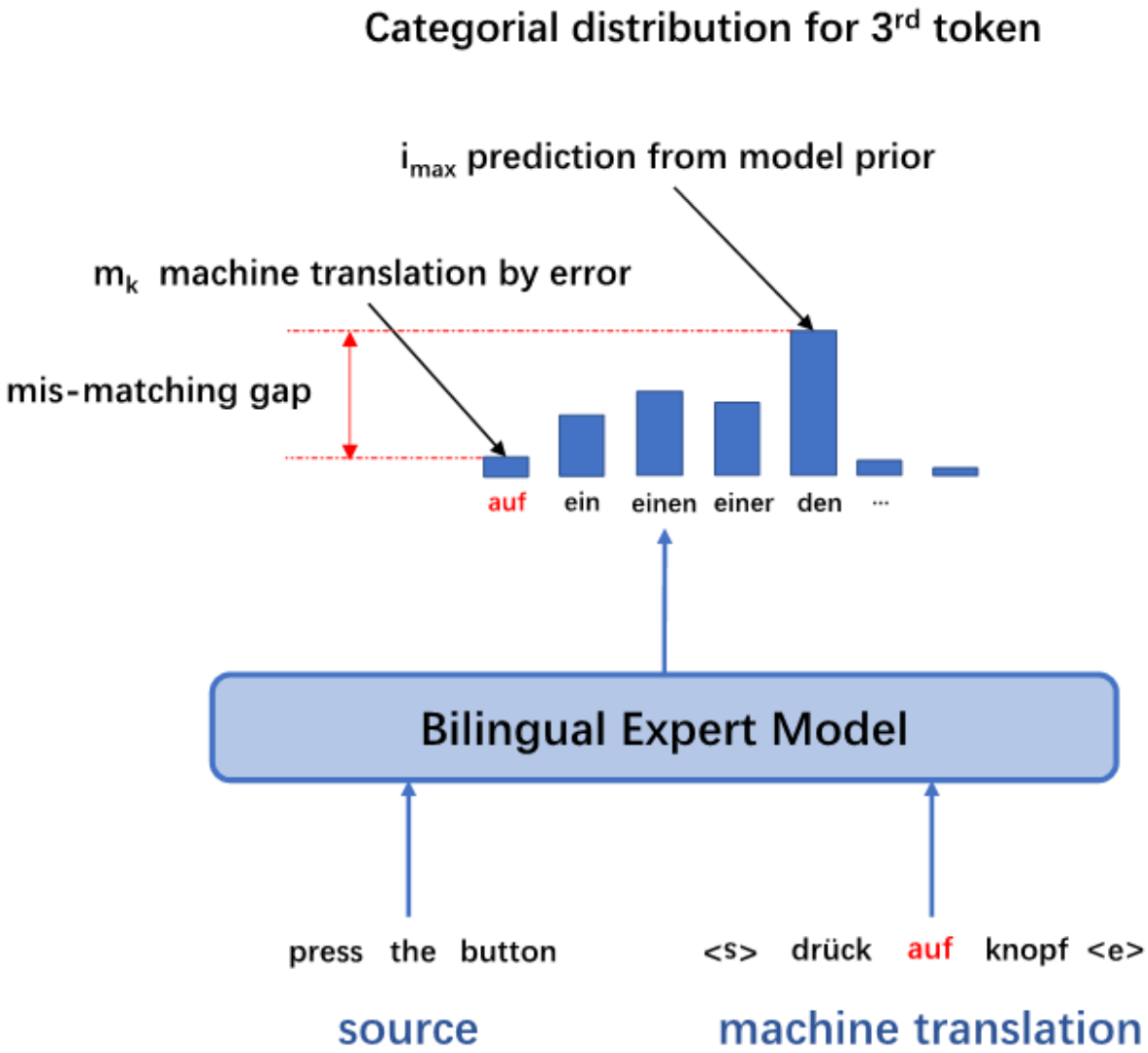
- CAT：computer-assisted translation， a form of language translation in which a human translator uses computer hardware to support and facilitate the translation process
- traditional direction for QE：formulate the sentence level score or word level tags prediction as a constraint regression or sequence la-beling problem respectively
- Another promising direction：build a multi-task learning model to i**ncorporate quality estimation task with automatic post-editing (APE) together**
- conditional language model：通过公开可获得的双语余料库可以轻松的训练出条件语言模型作为特征提取器，当原句与较差的翻译提供给该模型时，得出的特征分布**很可能**与正确的翻译不同
  - 除了特征学习，条件语言模型也可以进行APE
- four-dimensional token mis-matching features ： 从预训练的模型中提取出特征，这四维的不匹配特征能够反映the difference between **what the bilingual expert model will predic**t and the **actual token of machine translation output**
- 预训练策略的优点：可以改善**下游质量评估模型**的性能，即便用的是单层Bi-LSTM模型
- 阿里使用的预训练模型
  - **one transformer encoder for source sentence and a novel bidirectional transformer decoder for target sentence**
  - 与其它类似的模型最大的不同是transformer使用了双向的decoder
- 给定双语语料库，可以用公式表示一个机器翻译系统为 $p(t|s) = p(t|z) \times p(z|s)$
  - s代表原句的tokens sequence， t代表翻译的句子的tokens sequence，z(论文里表述为t**he latent variable to represent the encoded source sentence**, 感觉有点问题， 后文中提到z：generally summarize the information of the source and the target)， $p(z|s)$可以理解为encoder过程，$p(t|z)$可以理解为decoder过程
  - 训练数据集的形式为(s, m, t, h, y), t代表APE后的句子, h代表hter， y是二元向量代表机器翻译的句子是'OK'还是 'BAD'
  - 任务：learn a regression model p(h|s, m) and a sequence labeling model $p(y|s, m)$ $||a||_2^2$
- **Bilingual Expert Model**

- 根据贝叶斯公式可得， $$p(\mathbf{z}|\mathbf{t},\mathbf{s}) = \frac{p(\mathbf{t}|\mathbf{z})p(\mathbf{z}|\mathbf{s})}{p(\mathbf{t}|\mathbf{s})}$$ ，即用 src 与 tgt 来表示 z， 其中$p(t,z,s) = p(t|z) \times p(z|s)$
- p(t|s)是积分，不好计算，于是提出一个变量分布q(z|t,s)来近似的表示上述概率， 这个变量分布通过最小化K-L散度
$$\min D_{KL}(q(\mathbf{z}|\mathbf{t},\mathbf{s})\|p(\mathbf{z}|\mathbf{t},\mathbf{s}))$$
来计算
- 进一步转换为
$$\max \mathbb{E}_{q(\mathbf{z}|\mathbf{t},\mathbf{s})}[p(\mathbf{t}|\mathbf{z})] - D_{KL}(q(\mathbf{z}|\mathbf{t},\mathbf{s})\|p(\mathbf{z}|\mathbf{s}))$$
- 如果在优化过程中使用蒙特卡罗尔积分，那么上面公式的第一项可以考虑为一个**条件自编码系统(conditional auto-encoder system)**，如果把$p(z|s)$当做标准的高斯分布的话，那么上面公式的第二项可以被算出。如果忽略条件信息s的话，目标被简化为amortized variational inference or variational auto- encoders (VAE) framework
- 使用transformer构建上面的公式,
  - 总共有三个模块，**self-attention encoder** for the source sentence, **forward and backward self-attention** encoders for target sentence, and **the reconstructor** for the target sentence
  - 前两个模块代表$q(z|s,t)$, ~~即通过encoder、decoder过程训练特征~~$z = \{\overrightarrow{z}, \overleftarrow{z}\}$; 第三个模块代表$p(t|z)$~~,即根据提取出的特征构建出模型训练输出的翻译结果。~~(这里不能用等号，z是所有特征的抽象)
  - 假设有以下条件独立性成立
  $$p(\mathbf{t}|\mathbf{z}) = \prod_k p(t_k|\overrightarrow{\mathbf{z}_k}, \overleftarrow{\mathbf{z}_k})$$
  $$q(\mathbf{z}|\mathbf{s},\mathbf{t}) = \prod_k q(\overrightarrow{\mathbf{z}_k}|\mathbf{s},\mathbf{t}_{<k})q(\overleftarrow{\mathbf{z}_k}|\mathbf{s},\mathbf{t}_{>k})$$
  - 其中双向的隐含变量z包含所有的 $\{\overrightarrow{\mathbf{z}_k}, \overleftarrow{\mathbf{z}_k}\}$. 整个过程共用encoder，而decoder是双向的，由encoder到decoder有两个不同的z向量。
  - ELMO所用的方法(这篇论文参考了61篇论文🐱)： $\prod_k p(t_k|\overrightarrow{\mathbf{z}_k})p(t_k|\overleftarrow{\mathbf{z}_k})$ ，与阿里的不同之处在于划分的粒度更细，这里我的理解是这样的，ELMO的$\overrightarrow{z}$与$\overleftarrow{z}$是被孤立开来的，预测t_k是独立进行的，最后把预测的结果合并，而阿里的方法是先做$\overrightarrow{z}$与$\overleftarrow{z}$的合并，之后再用合并的结果预测t_k
  - 根据第二个公式，$\{\overrightarrow{z_k}, \overleftarrow{z_k}\}$来自于$q(\overrightarrow{z_k}|s,t_{<k})$、$q(\overleftarrow{z_k}|s,t_{>k})$, 假设上述两项服从高斯分布$q(\cdot|\cdot) \sim N(\mu(s,t), \sigma^2 I)$，并且$\mu(s,t)$的计算只依赖于当前的s, t ， 那么通过控制超参数σ可以增加z的不确定性，从而防止数据过度拟合. σ小一点比较好.
- 另外，阿里使用的bidirectional self-attention transformer并不是生成模型，所以不能称之为decoder。d但为了方便，还是称之为decoder吧！
- 生成模型：https://www.zhihu.com/question/20446337

======论文中以上内容均未正式提到模型的特征=====

======下面才是特征的内容=======

- Model Derived Features， 需要从模型中提取出来的特征包括($\overrightarrow{z_k}$、$\overleftarrow{z_k}$、 $\overrightarrow{e_{t_{k-1}}}$、$\overleftarrow{e_{t_{k+1}}}$)
  - 模型的输入为：(s, m)
  - **从bilingual expert model提取的特征之后会用于预测翻译错误**
  - **Naturally**, 关 于 $\overrightarrow{z_k}$、$\overleftarrow{z_k}$ ：
    $$q(\mathbf{z}|\mathbf{s},\mathbf{t}) = \prod_k q(\overrightarrow{\mathbf{z}_k}|\mathbf{s},\mathbf{t}_{<k})q(\overleftarrow{\mathbf{z}_k}|\mathbf{s},\mathbf{t}_{>k})$$ 这里是故意这么做的，对于正向decoder，$\overrightarrow{z_k}$能够包含原句s以及目标句第k个词之前的信息，而对于反向decoder而言，$\overleftarrow{z_k}$能够包含原句s以及目标句第k个词之后的信息.。**这两个特征有助于之后手动提取mis-matching features。**
  - 关于$\overrightarrow{e_{t_{k-1}}}$、$\overleftarrow{e_{t_{k+1}}}$ ：把bilingual expert model的输出作为输出再次训练(这里可能理解错了，模型的输出可能被送往下游的模型而不是自身)，这里借鉴了ELMO， 特征为token embedding feature， 具体是$\overrightarrow{e_{t_{k-1}}}$、$\overleftarrow{e_{t_{k+1}}}$(第k-1个词的词向量，第k+1个词的词向量)
  - 不考虑t_k作为特征是因为t_k中有错误，不够准确
- Mis-matching Features(这里的内容与上面有重复)
  - 上面提到的Model Derived Features是计算图中的节点, 除此之外还有一些非常重要的特征即Mis-matching Features, 它们能够反映billingual expert model预测的输出与真实的输出之间的差异。
  -



)

- 对于bilingual expert model 理论上有$p(m_k|\cdot) \leq p(t_k|\cdot)$ for optimal model if $m_k \neq t_k$，也就是说模型预测出正确的词的概率最大， 于是可定义如下四维的不匹配特征
  - $$\mathbf{f}_k^{mm} = (\mathbf{l}_{k,m_k}, \mathbf{l}_{k,i_{\max}^k}, \mathbf{l}_{k,m_k} - \mathbf{l}_{k,i_{\max}^k}, \mathbb{I}_{m_k \neq i_{\max}^k})$$
  - 具体参考上文mis-matching特征，直观上可以理解为(1. 正确答案，2. 模型预测答案，1.与2.的差异，1.与2.是否不等)，如果丢掉前两个维度，只保留最后两个维度会怎样？
  - 如果模型预测正确，那么1.与2.相等，第三、第四均为0

=======下面是质量评估模型=======

- Bi-LSTM Quality Estimation
  - 没有使用transformer的原因是效果较差
  - 把之前提取的所有特征组合成为单个向量，记为$\{f_k\}_{k=1}^T$, T : the number of tokens in m, k的范围为1-T，从这里可以看出**特征向量的的数量等于m中单词的数量，那么可以简单的进行合并定义** $\{f_k\}_{k=1}^T = \{\overrightarrow{z_k}, \overleftarrow{z_k}, \overrightarrow{e_{t_{k-1}}}, \overleftarrow{e_{t_{k-1}}}, f_k^{mm}\}$, 句子级别的hter预测可以通过(9), 单词级别的错误性预测可以通过(10)来完成

-

$$\overrightarrow{\mathbf{h}_{1:T}}, \overleftarrow{\mathbf{h}_{1:T}} = \text{Bi-LSTM}(\{\mathbf{f}_k\}_{k=1}^{T}) \tag{8}$$

$$\arg\min \left\| h - \text{sigmoid}\left(\mathbf{w}^{\top}[\overrightarrow{\mathbf{h}_T}, \overleftarrow{\mathbf{h}_T}]\right) \right\|_2^2 \tag{9}$$

$$\arg\min \sum_{k=1}^{T} \text{XENT}(y_k, \mathbf{W}[\overrightarrow{\mathbf{h}_k}, \overleftarrow{\mathbf{h}_k}]) \tag{10}$$

- w为一个向量，W为矩阵，y_k : the error **label** for the k-th token of translation output，XENT为cross entropy loss (with logits)， h的范围为[0-1]， 对于(9)而言，$[\overrightarrow{h_T}, \overleftarrow{h_T}]$, 为bi-LSTM最后一个时刻的输出，能够反映整个句子的信息。对于(10)而言，由于是单词级别的预测，因而需要对每一个$[\overrightarrow{h_k}, \overleftarrow{h_k}]$作运算, 运算的结果与$y_k$求交叉熵，最小化该损失函数。
-

Algorithm 1 Translation Quality Estimation with Bi-Transformer and Bi-LSTM

Require: QE training data $(\mathbf{s}, \mathbf{m}, \mathbf{t}, h, \mathbf{y})_{1:M}$, QE inference data $(\mathbf{s}, \mathbf{m})$, and parallel corpus $(\mathbf{s}, \mathbf{t})_{1:N}$.

1: Combine the parallel corpus with 10 copies of QE training parallel corpus $C = (\mathbf{s}_n, \mathbf{t}_n)_{n=1}^{N} \bigcup 10 \times (\mathbf{s}_m, \mathbf{t}_m)_{m=1}^{M}$

2: Pre-train bilingual expert model via the bidirectional transformer on the combined corpus C.

3: Extract features $\mathbf{f}_k = \text{Concat}(\overrightarrow{\mathbf{z}_k}, \overleftarrow{\mathbf{z}_k}, \mathbf{e}_{t_{k-1}}, \mathbf{e}_{t_{k+1}}, \mathbf{f}_k^{mm})$ for QE training data $(\mathbf{s}, \mathbf{m})$.

4: Train Bi-LSTM model via objectives (9)(10).

5: **return** Predict $h, \mathbf{y}$ for QE inference data

- 第三步， $f_k = Concat(\overrightarrow{z_k}, \overleftarrow{z_k}, \overrightarrow{e_{t_{k-1}}}, \overleftarrow{e_{t_{k-1}}}, f_k^{mm})$

  =====剩余的内容===

  - To ensure the quality of the corpora, we filtered the source and target sentence with length ≤70 and the length ratio between 1/3 to 3, thus resulting roughly 9 million (2017) and 25 million (2018) parallel sentences pairs for both English↔German directions。
  - 使用了BPE tokenization来减少未知tokens的数量
  - 在word token tagging prediction and BPE tokenization之间存在一些差异，需要解决
  - 双向transformer中encoder、 decoder模块数量是2， hidden units for feed-forward sub-layer is 512、 8-head self-attention、
  - Sentence Level Scoring And Ranking：
    - The primary metrics of sentence level task are Pearson's correlation and Spearman's rank correlation of the entire testing data
    - model derived features (MD) and mis-matching features(MM)：仅仅使用MM，也可以取得非常好的效果

- Word Level For Word Tagging
  - The metric of word level is evaluated in terms of classification performance via the multiplication of F1-scores for the 'OK' and 'BAD' classes against the true labels
  - For the binary classification, we tuned the threshold of the classifier on the development data and applied to the test data. 在开发数据集上调整了分类器的阈值并应用于测试数据。原因是算法偏向于把word分类为good，因为true lables非常不平衡。
  - The higher value of single F1-OK or F1-BAD cannot reflect the robustness of the algorithm, since it may result in lower F1 of another metric.
- Word Level For Gap Tagging(有问题)
  - $$\arg\min \sum_{k=0}^{T} \mathrm{XENT}(g_k, \mathbf{W}[\overrightarrow{\mathbf{h}_k}, \overleftarrow{\mathbf{h}_k}, \overrightarrow{\mathbf{h}_{k+1}}, \overleftarrow{\mathbf{h}_{k+1}}]) \quad (11)$$

    g_k为第k个和第k+1个标签之间的gap tag， 这个公式足够用来进行gap prediction, 这里的预测仅仅是分类为OK、BAD标签， 而下面的公式可以用来进行APE
  - 具体建模如下
  - $$p(\mathbf{t}, \mathbf{t}^g|\mathbf{z}) = p(\mathbf{t}|\mathbf{z})p(\mathbf{t}^g|\mathbf{z}) \quad q(\mathbf{z}|\mathbf{s}, \mathbf{m}),$$
  - cussed model, gap token prediction distribution $p(\mathbf{t}^g|\mathbf{z}) = \prod_k p(t_k^g|\overrightarrow{\mathbf{z}_k}, \overleftarrow{\mathbf{z}_k}, \overrightarrow{\mathbf{z}_{k+1}}, \overleftarrow{\mathbf{z}_{k+1}})$ and q becomes conditional on
  - $q(z|s,m)$, $q(t|z)$都是此前训练好的模型， q此时是m的条件，为了进行gap prediction需要定义 token表示nothing needs to be inserted
- Extending to BPE Tokenization
  - 使用BPE可以很好的处理出现次数非常少的words, 比如一些复合词
  - 对于句子级别的HTER预测而言，使用BPE不会产生什么冲突，原因：回归模型仅仅关注最后时刻的输出状态
  - 在单词级别上进行标注会出现问题，这应该算是编码的问题，BPE tokenization之后序列的长度特征L_b将**不同**于word tokens的数量L_w， 解决方法为：average the features of all subword units belonging to one single word token， 横轴为L_b, 纵轴为L_w, 可以通过矩阵乘法来计算平均特征。下图纵轴为原序列，横轴为BPE tokneization后的序列，横轴之和为1，纵轴为模型的input

	_the	_class room	_is	_s	mall
the	1	0	0	0	0	0
classroom	0	1/2	1/2	0	0	0
is	0	0	0	1	0	0
small	0	0	0	0	1/2	1/2

  - 细粒度的BPE tokenization可以提高大多数任务的QE性能