用于同步机器翻译的隐马尔可夫 Transformer

Shaolei Zhang [1,2], **Yang Feng** [1,2]*1智能信息处理重点实验室，中国科学院计算技术研究所（ICT/CAS）2中国科学院大学，中国北京{zhangshaolei20z,fengyang}@ict.ac.cn

arXiv:2303.00257v1 [cs.CL] 1 Mar 2023

摘要

同步机器翻译（SiMT）在接收源序列的同时输出目标序列，因此，学习何时开始翻译每个目标词元是 SiMT 任务的核心挑战。然而，在众多可能的开始翻译时刻中学习最优时刻并非易事，因为开始翻译的时刻总是隐藏在模型内部，只能通过观测目标序列进行监督。在本文中，我们提出了隐马尔可夫 *Transformer*（*HMT*），将开始翻译的时刻视为隐事件，将目标序列视为相应的观测事件，从而把二者组织为隐马尔可夫模型。HMT 显式地将多个开始翻译的时刻建模为候选隐事件，然后选择其中一个来生成目标词元。在训练过程中，通过最大化多个开始翻译时刻下目标序列的边缘似然，HMT 学习在能够更准确地生成目标词元的时刻开始翻译。在多个 SiMT基准测试上的实验表明，HMT 超越了强大的基线方法，并取得了最先进的性能[1]。

1 引言

近年来，随着直播、视频字幕和会议等实时场景不断增加，同时机器翻译（SiMT）受到越来越多的关注（Cho & Esipova, 2016; Gu 等人, 2017; Ma 等，2019; Arivazhagan 等，2019）。这类任务要求模型逐个接收源端词元，同时生成目标词元。为了在低延迟下实现高质量翻译，同时机器翻译模型需要学会何时开始翻译每个目标词元（Gu 等人, 2017），从而在翻译过程中，在等待下一个源端词元（即读取动作）和生成一个目标词元（即写入动作）之间做出明智的决策。

然而，同时机器翻译模型要学会何时开始翻译目标词元并非易事，因为开始翻译的时刻始终隐含在模型内部，而我们只能用观测目标序列监督同时机器翻译模型（Zhang & Feng, 2022a）。现有的同时机器翻译方法在决定何时开始翻译方面分为固定方法和自适应方法。固定方法依据预定义规则直接决定何时开始翻译，而非通过学习得到这些规则（Dalvi 等人, 2018; Ma 等，2019; Elbayad 等，2020）。这类方法忽略上下文，因此有时即使源端内容不足，也会强迫模型开始翻译（Zheng 等，2020a）。自适应方法会动态决定读取/写入动作，例如预测一个变量来指示读取/写入动作（Arivazhagan 等，2019; Ma 等，2020; Miao 等，2021）。然而，由于读取/写入动作与观测目标序列之间缺乏明确的对应关系（Zhang & Feng, 2022c），仅凭观测目标序列的监督很难学到精确的读取/写入动作（Alinejad 等人, 2021; Zhang & Feng, 2022a; Indurthi 等人, 2022）。

为了找到模型内部每个目标词元开始翻译的最佳时机，理想的解决方案是将开始翻译的时机与观测到的目标

* 通讯作者：Yang Feng。[1] 代码可在 https://github.com/ictnlp/HMT 获取

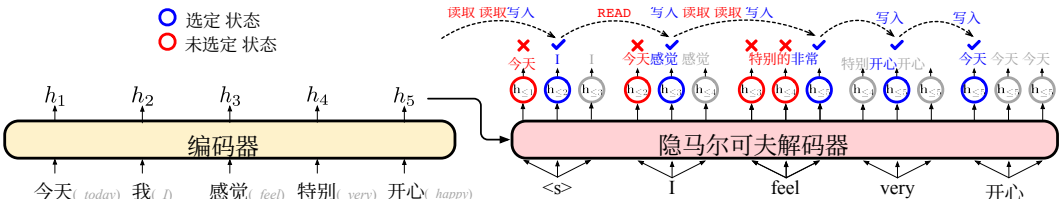


图 1：隐马尔可夫 Transformer 示意图。HMT 会为每个目标词元显式生成 $K = 3$ 个状态，分别表示在接收到不同数量的源词元时开始翻译该目标词元（其中 $\mathbf{h}_{\leq n}$ 表示接收到前 n 个源词元时开始翻译）。随后，HMT 从低延迟到高延迟（即从左到右）判断是否选取每个状态。一旦选定某个状态，HMT 就会根据选定的状态翻译目标词元。

序列明确对应起来，并进一步学会在能够更准确地生成目标词元的时机开始翻译。为此，我们提出了用于同时机器翻译（SiMT）的隐马尔可夫 *Transformer*（*HMT*），将开始翻译的时机视为隐事件，将翻译结果视为相应的观测事件，从而以隐马尔可夫模型的形式对其进行组织（Baum 和 Petrie，1966；Rabiner 与 Juang，1986；Wang 等人，2018）。如图 1 所示，HMT 为每个目标词元显式地产生一组状态，其中多个状态分别表示在不同时间开始翻译该目标词元（即接收不同数量的源词元后开始翻译）。随后，HMT 从低延迟到高延迟依次判断是否选择各个状态。一旦选定某个状态，HMT 就会基于该状态生成目标词元。例如，在生成第一个目标词元时，HMT 会产生 3 个状态，分别表示接收第 1、2 和 3 个源词元后开始翻译（即 $\mathbf{h}_{\leq 1}$、$\mathbf{h}_{\leq 2}$ 和 $\mathbf{h}_{\leq 3}$）。随后在判断过程中，第一个状态未被选中，第二个状态被选中并输出“*I*”，之后不再考虑第三个状态，因此 HMT 在接收到前 2 个源词元时开始翻译“*I*”。训练过程中，HMT 通过在所有可能的选择结果（即隐事件）上最大化目标序列（即观测事件）的边缘似然来进行优化。这样，更准确地生成目标词元的状态（时机）就更有可能被选中，从而使 HMT 能够在观测目标序列的监督下有效学会何时开始翻译。在英语→越南语和德语→英语 SiMT基准测试上的实验表明，HMT 在所有延迟水平下均优于强基线，并取得最先进的性能。

2 相关工作

学习何时开始翻译是同时机器翻译的关键。近期的同时机器翻译方法可分为固定方法和自适应方法。对于固定方法，Ma 等（2019）提出了 wait-k 策略，该策略首先读取 k 个源词元，然后交替读取/写入一个词元。Elbayad 等（2020）提出了一种针对 wait-k 策略的高效训练方法，该方法会在批次间随机抽取不同的 k。Zhang 与 Feng（2021a）提出了字符级 wait-k 策略。Zheng 等（2020a）提出了自适应 Wait-k，在推理过程中以启发式方式整合多个 Wait-k模型。Guo 等（2022）提出了针对 wait-k 策略的后评估方法。Zhang 等（2022）提出了等待信息策略，通过量化词元信息来改进 wait-k 策略。Zhang 与 Feng （ 2021c ）提出了混合专家 Wait-k，通过多个专家学习多个 wait-k 策略。对于自适应方法，Gu 等（2017）通过强化学习训练了一个 READ/WRITE 智能体。Zheng 等（2019b）使用规则生成的黄金动作训练该智能体。Zhang 与 Feng（2022c）提出了 GMA，根据预测对齐来决定何时开始翻译。Arivazhagan 等（2019）提出了 MILk，该方法使用一个变量表示 READ/WRITE，并将该变量与单调注意力联合训练（Raffel 等人，2017）。Ma 等（2020）提出了 MMA，在 Transformer 上实现 MILk。Liu 等（2021）提出了用于同时机器翻译的 CAAT，将 RNN-T 引入 Transformer 架构。Miao 等（2021）提出了 GSiMT 来生成 READ/WRITE 动作；该方法在训练期间隐式考虑所有读取/写入组合，并在推理时选取其中一种组合。Zhang 与 Feng（2022d）提出了一种用于同时机器翻译的基于信息传输的策略。

与以往方法相比，HMT 在训练和推理过程中都显式地建模多个可能的开始翻译时刻，并通过隐马尔可夫模型将同时机器翻译中的两个关键问题——“学习何时开始翻译”和“学习翻译”——整合到一个统一框架中。

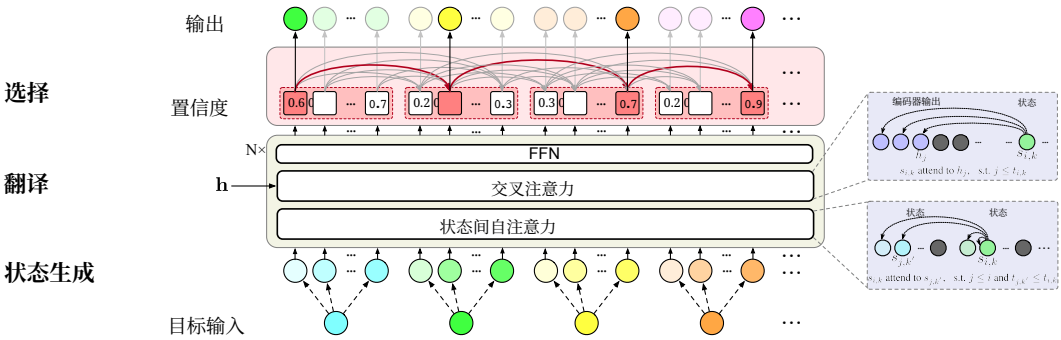


图 2：隐马尔可夫解码器的架构。

3 隐马尔可夫 Transformer

为学习同时机器翻译模型中隐藏的最优开始翻译时刻，我们提出了隐马尔可夫 Transformer（HMT），以隐马尔可夫模型的形式组织“开始翻译时刻”和“观测目标序列”。通过最大化观测目标序列在多个可能的开始翻译时刻上的边缘似然，HMT 学会何时开始翻译。下面将介绍 HMT 的架构、训练和推理。

3.1 架构

隐马尔可夫 Transformer（HMT）由一个编码器和一个隐马尔可夫解码器组成。将源序列记为 $\mathbf{x} = (x_1, \cdots, x_J)$，目标序列记为 $\mathbf{y} = (y_1, \cdots, y_I)$，采用单向编码器（Arivazhagan 等，2019；Ma 等，2020；Miao 等，2021）将 $\mathbf{x}$ 映射为源端隐藏状态 $\mathbf{h} = (h_1, \cdots, h_J)$。隐马尔可夫解码器会明确生成一组状态 y_i，分别对应从多个翻译时刻开始翻译的情况，然后判断选择哪个状态来输出目标词元。具体而言，如图 2 所示，隐马尔可夫解码器包含三个部分：状态生成、翻译和选择。

状态生成 翻译 y_i时，HMT 首先生成一组 K 状态 $\mathbf{s}_i = \{s_{i,1}, \cdots, s_{i,K}\}$，分别表示接收到不同数量的源词元时开始翻译 y_i。然后，我们为这些状态设定翻译时刻 $\mathbf{t}_i = \{t_{i,1}, \cdots, t_{i,K}\}$，其中状态 $s_{i,k}$ 要求在接收到前 $t_{i,k}$ 个源词元 $\mathbf{x}_{\leq t_{i,k}}$时开始翻译。

为了为状态设置合适的$\mathbf{t}$，我们旨在预先剪枝掉那些不利的翻译时刻，例如在接收 x_J 后翻译 y_1，或在接收 x_1 后翻译 y_I（Zhang 和 Feng，2022b；a）。因此，如图 3 所示，我们相应地引入 wait-L 路径作为下界，并引入 wait-$(L+K-1)$ 路径作为上界（Zheng 等人，2019a），然后考虑在此区间内开始翻译的合适时刻，其中 L 和 K 是超参数。形式化地，状态 $s_{i,k}$ 的翻译时刻$t_{i,k}$ 定义为：

$$t_{i,k} = \min\left\{L + (i-1) + (k-1), |\mathbf{x}|\right\}. \tag{1}$$

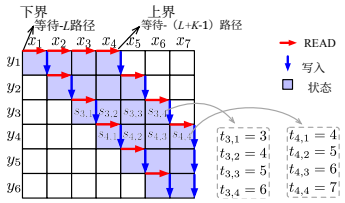


图 3：翻译时刻的设置 $\mathbf{t}$（例如，$L=1$、$K=4$）。

翻译 每个目标词元的 K 状态表示通过将目标输入上采样 K 倍进行初始化，然后经过 N 个 Transformer 解码器层计算得到，每个解码器层由三个子层组成：状态间自注意力、交叉注意力和前馈网络。

对于状态间的自注意力，状态 $s_{i,\,k}$ 可以关注所有先前目标词元的状态 $s_{j,\,k'}$（即 $j \leq i$），同时确保 $s_{j,\,k'}$ 最迟在 $s_{i,\,k}$ 开始翻译（即 $t_{j,\,k'} \leq t_{i,\,k}$），以避免未来的源信息从$s_{j,\,k'}$ 泄漏到 $s_{i,\,k}$。从 $s_{i,\,k}$ 到 $s_{j,\,k'}$ 的自注意力为：

$$\text{SelfAtt}\left(s_{i,k}, s_{j,k'}\right) = \begin{cases} \text{softmax}\left(\frac{s_{i,k}\mathbf{W}^Q \left(s_{j,k'}\mathbf{W}^K\right)^\top}{\sqrt{d}}\right) & \text{if } j \leq i \text{ and } t_{j,k'} \leq t_{i,k} \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

其中，$\mathbf{W}^Q$ 和$\mathbf{W}^K$ 是可学习参数。得益于状态间的自注意力，HMT通过考虑不同的开始翻译时刻，可以捕获更全面的状态表示。对于交叉注意力，由于状态 $s_{i,\ k}$ 在接收到第一个 $t_{i,\ k}$ 源词元时开始翻译，$s_{i,\ k}$ 只能关注源端隐藏状态 h_j 及其 $j \leq t_{i,\ k}$，计算如下：

$$\mathrm{CrossAtt}\left(s_{i,k}, h_j\right) = \begin{cases} \mathrm{softmax}\left(\frac{s_{i,k}\mathbf{W}^Q\left(h_j\mathbf{W}^K\right)^\top}{\sqrt{d}}\right) & \text{if } j \leq t_{i,k} \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

经过 N 解码器层后，我们得到状态 $s_{i的k}$最终表示。因此， y_i 从该状态$s_{i生成的k}$ 翻译概率根据其最终表示计算：

$$p\left(y_i \mid \mathbf{x}_{\leq t_{i,k}}, \mathbf{y}_{<i}\right) = \mathrm{softmax}\left(s_{i,k}\mathbf{W}^O\right), \tag{4}$$

其中，$\mathbf{W}^O$ 是可学习参数，$\mathbf{y}_{<i}$ 是之前的目标词元。

选择获得状态的最终表示后，为了判断是否选择状态$s_{i来k}$ 生成 y_i，HMT 会预测选择状态 $s_{i的k}$置信度 $c_{i,\ k}$ 。该置信度$c_{i,\ k}$ 根据$s_{i的k}$ 最终状态表示以及已接收的源内容进行预测：

$$c_{i,k} = \mathrm{sigmoid}\left(\left[\overline{\mathbf{h}}_{\leq t_{i,k}} : s_{i,k}\right]\mathbf{W}^S\right), \tag{5}$$

其中，$\overline{\mathbf{h}}_{\leq t_{i,k}} = \frac{1}{t_{i,k}} \sum_{j=1}^{t_{i,k}} h_j$ 是对已接收源词元的隐藏状态进行平均池化的结果， [:] 是拼接操作，$\mathbf{W}^S$ 是可学习参数。在推理过程中，HMT 判断是否从 $s_{i,1}$ 到 $s_{i,K}$中选择状态。如果 $c_{i,\ k} \geq 0.5$ （即达到足够的置信度），HMT 选择状态 $s_{i,k}$ 来生成 y_i。否则，HMT 移动到下一个状态 $s_{i,k+1}$ 并重复判断。注意，我们将 $c_{i,K} = 1$ 设置为确保 HMT 在最后一个状态 s_i之前开始翻译。

3.2 训练

由于开始翻译的时刻隐藏在模型中，而目标序列是可观测的，HMT 将何时开始翻译目标词元（即选择了哪些状态）视为隐事件，并将目标词元视为观测事件。此外，HMT 以隐马尔可夫模型的形式组织这些事件，从而将开始翻译的时刻与观测到的目标词元关联起来。形式上，对于隐事件，我们将选中的状态表示为 $\mathbf{z} = (z_1, \cdots, z_I)$，其中 $z_i \in [1, K]$ 表示选择状态 s_{i,z_i} 来生成 y_i。接着，按照 HMM 的形式，我们引入选择之间的**转移概率** ，以及从选择结果到目标词元的**发射概率**。

转移概率 表示在前一次选择 z_{i-1}的条件下进行选择 z_i 的概率，记作 $p(z_i \mid z_{i-1})$。由于 HMT 判断是否选择从 $s_{i,1}$ 到 $s_{i,K}$ 的状态（即从低延迟到高延迟），只有在前一个状态 $s_{i,k}$ 未被选中时，才能选择 $s_{i,k+1}$ 。因此， $p(z_i \mid z_{i-1})$的计算分为两部分[2]：**(1)** s_{i,z_i} 有较高的把握被选中；**(2)** 翻译时刻位于$t_{i-1,z_{i-1}}$ 和 t_{i,z_i} 之间的那些状态（即在s_{i,z_i}之前已判定的状态）没有较高的把握被选中，计算如下[3]：

$$p(z_i \mid z_{i-1}) = \begin{cases} c_{i,z_i} \times \prod\limits_{t_{i-1,z_{i-1}} \leq t_{i,l} < t_{i,z_i}} (1 - c_{i,l}) & \text{if } t_{i,z_i} \geq t_{i-1,z_{i-1}} \\ 0 & \text{if } t_{i,z_i} < t_{i-1,z_{i-1}} \end{cases}. \tag{6}$$

发射概率表示在选定状态s_i下观测到目标词元 $y_{i\ z_i}$的概率，即式（4）中的翻译概率。为清晰起见，我们将其改写为 $p\left(y_i \mid \mathbf{x}_{\leq t_{i,z_i}}, \mathbf{y}_{<i}, z_i\right)$，以强调 HMT 在选择 z_i下生成目标词元 y_i 的概率。

HMM 损失为了学习何时开始翻译，我们通过最大化目标序列（即观测事件）在所有可能的选择结果（即隐事件）上的边缘似然来训练 HMT，从而使 HMT 对能够更准确地生成目标词元的状态赋予更高的置信度。给定转移概率和发射概率，观测目标序列 $\mathbf{y}$ 在所有可能的选择结果$\mathbf{z}$ 上的边缘似然计算如下：

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}) \times p(\mathbf{z}) = \sum_{\mathbf{z}} \prod_{i=1}^{|\mathbf{y}|} p\left(y_i \mid \mathbf{x}_{\leq t_{i,z_i}}, \mathbf{y}_{<i}, z_i\right) \times p(z_i \mid z_{i-1}). \tag{7}$$

[2]有关选择之间转移概率的详细说明，请参见附录 A。[3]我们在 z_1 之前添加了一个选择 z_0，其具有 p（z_0）$= 1$，以表示在翻译开始时未接收到任何源词元，即 $t_{0,z_0} = 0$。因此， p（$z_1 \mid z_0$）是 HMT 中该选择的初始概率。

算法1隐马尔可夫 Transformer 的推理策略

输入： 源序列 $\mathbf{x}$；翻译时刻$\mathbf{t}$；状态数 K。
输出： 译文序列 $\hat{\mathbf{y}}$。
初始化： 接收到的源序列 $\hat{\mathbf{x}}=[]$，源索引 $j=0$；译文序列 $\hat{\mathbf{y}}=[\langle\text{bos}\rangle]$，目标索引 $i=1$。

1: **当** $\hat{y}_{i-1} \neq \langle\text{eos}\rangle$ **时**
2: **如果** $\lceil \hat{\mathbf{x}} \rceil < t_{i,1}$ **那么** ▷到达下界
3: 等待源词元，直到 $\lceil \hat{\mathbf{x}} \rceil = t_{i,1}$；$j \leftarrow t_{i,1}$；
4: **对于** $k \leftarrow 1$ 到 K **执行**
5: **如果** $\lceil \hat{\mathbf{x}} \rceil > t_{i,k}$ **那么继续**； ▷ 若其翻译时刻$s_{i,k}$ 小于 $\lceil \hat{\mathbf{x}} \rceil$，则跳过$t_{i,k}$
6: 根据式（5）计算状态表示$s_{i,k}$ 及其置信度$c_{i,k}$；
7: **若** $c_{i,k} \geq 0.5$ **那么** ▷ 选择 ⇒ 写入
8: 根据式（4）翻译目标词元$\hat{y}_i$；
9: $\hat{\mathbf{y}} \leftarrow \hat{\mathbf{y}} + \hat{y}_i$； $i \leftarrow i+1$；
10: **中断**；
11: **else** ▷ Unselect ⇒ READ
12: 等待下一个源端词元 x_{j+1}；
13: $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + x_{j+1}$； $j \leftarrow j+1$；
14: **返回**$\hat{\mathbf{y}}$；

我们采用动态规划来降低对所有可能的选择结果进行边缘化的计算复杂度，详细计算过程请参阅附录 B。随后，使用负对数似然损失 $\mathcal{L}_{hmm}$对 HMT 进行优化：

$$\mathcal{L}_{hmm} = -\log p(\mathbf{y} \mid \mathbf{x}) \,. \tag{8}$$

延迟损失 此外，我们还引入延迟损失 $\mathcal{L}_{latency}$，以权衡翻译质量与延迟。$\mathcal{L}_{latency}$ 也通过对所有可能的选择结果$\mathbf{z}$进行边缘化来计算：

$$\mathcal{L}_{latency} = \sum_{\mathbf{z}} p(\mathbf{z}) \times \mathcal{C}(\mathbf{z}) = \sum_{\mathbf{z}} \prod_{i=1}^{|\mathbf{y}|} p(z_i \mid z_{i-1}) \times \mathcal{C}(\mathbf{z}) \,, \tag{9}$$

其中，$\mathcal{C}(\mathbf{z})$ 是用于衡量给定选择结果$\mathbf{z}$延迟的函数，其计算方式是相对于下界采用平均滞后（Ma 等，2019）：$\mathcal{C}(\mathbf{z}) = \frac{1}{|\mathbf{z}|} \sum_{i=1}^{|\mathbf{z}|} (t_{i,\,z_i} - t_{i,\,1})$。

状态损失 我们还引入状态损失 $\mathcal{L}_{state}$，以促使 HMT 无论选择哪个状态 z_i（即无论何时开始翻译），都能生成正确的目标词元 y_i，从而提高选择的鲁棒性。由于训练期间会将 K 个状态并行输入隐马尔可夫解码器，$\mathcal{L}_{state}$ 通过对所有状态计算交叉熵损失得出：

$$\mathcal{L}_{state} = -\frac{1}{K} \sum_{i=1}^{|\mathbf{y}|} \sum_{z_i=1}^{K} \log p\big(y_i \mid \mathbf{x}_{\leq t_{i,z_i}}, \mathbf{y}_{<i}, z_i\big) \,. \tag{10}$$

最后，HMT 的总损失 $\mathcal{L}$ 计算如下：

$$\mathcal{L} = \mathcal{L}_{hmm} + \lambda_{latency}\mathcal{L}_{latency} + \lambda_{state}\mathcal{L}_{state}, \tag{11}$$

其中，我们在实验中设定 $\lambda_{latency} = 1$和 $\lambda_{state} = 1$。

3.3 推理

在推理时，HMT 根据置信度判断是否选择各个状态，顺序从低延迟到高延迟；一旦选定某个状态，HMT 就根据状态表示生成目标词元。具体而言，将当前已接收的源词元记为 $\hat{\mathbf{x}}$，HMT 的推理策略如算法1所示。翻译 $\hat{y}_i$ 时，HMT 会判断是否选择从 $s_{i,1}$ 到 $s_{i,K}$ 的状态，因此 HMT 至少会等待 $t_{i,1}$ 个源词元，以到达下界（第2行）。判断过程中，如果置信度 $c_{i,k} \geq 0.5$，HMT 就会根据式(4)选择状态 $s_{i,k}$ 来生成 $\hat{y}_i$（即执行写入动作）；否则，HMT 会等待下一个源词元（即执行读取动作），并转到下一个状态 $s_{i,k+1}$。通过设置 $c_{i,K}=1$，我们确保 HMT 在最后一个状态 $s_{i,K}$ 之前开始翻译 $\hat{y}_i$（即在到达上界之前）。注意，由于同时机器翻译中各翻译时刻的单调性，如果状态 $s_{i,k}$ 的翻译时刻 $t_{i,k}$ 小于已接收源词元的数量 $\lceil \hat{\mathbf{x}} \rceil$（即翻译 $\hat{y}_{i-1}$的时刻），则会跳过该状态，无法将其选定（第5行）；这与训练期间式(6)中选择之间的转移一致。

4 实验

4.1 数据集

我们在以下数据集上进行了实验，这些数据集是广泛使用的同时机器翻译基准测试。

IWSLT15[4] **英语→越南语（En→Vi）**（133K 对）我们使用 TED tst2012（1553 对）作为验证集，使用 TED tst2013（1268 对）作为测试集。遵循先前的设置（Ma 等，2020；Zhang 和 Feng，2021c），我们将词频低于 5 的词元替换为 $\langle unk \rangle$，英语和越南语的词汇量分别为 17K 和 7.7K。

WMT15[5] **德语 → 英语（De→En）**（450 万对）我们使用 newstest2013（3000 对）作为验证集，使用 newstest2015（2169 对）作为测试集。采用字节对编码（Sennrich 等，2016），执行 32K 次合并操作，并共享德语和英语的词汇表。

4.2 实验设置

我们针对几种表现出色的同时机器翻译方法进行了实验，具体如下。

整句机器翻译（Vaswani 等，2017）Transformer 模型会等待源序列全部完成后才开始翻译；我们还使用单向编码器进行对比。

Wait-k（Ma 等，2019）Wait-k 策略首先等待 k 个源词元，然后交替执行翻译一个目标词元和等待一个源词元。

多路径 Wait-k（Elbayad 等，2020）多路径 Wait-k 通过在不同批次间随机采样不同的 k 来训练 Wait-k 模型，并在推理时使用固定的 k。

自适应 Wait-k（Zheng 等，2020a）自适应 Wait-k 会训练一组 Wait-k 模型（例如，从 Wait-1 到 Wait-13），并在推理时以启发式方式组合这些模型。

MoE Wait-k[6] **(Zhang 和 Feng，2021c)** 混合专家（MoE）Wait-k 分别应用多个专家来执行 o lear多种 Wait-k 策略，并且在推理时也使用固定的 k。

MMA[7] **(Ma 等，2020)** 单调多头注意力（MMA）会预测一个变量，用于指示 e RE READ/WRITE 操作，并通过单调注意力训练该变量（Raffel 等人，2017）。

GMA[8] **(Zhang & Feng，2022c)** 高斯多头注意力（GMA）引入高斯先验来学习注意力中的对齐，并根据对齐位置决定何时开始翻译。**GSiMT(Miao 等，2021)** 生成式同时机器翻译（GSiMT）会生成一个变量来指示 READ/WRITE 操作。GSiMT 在训练期间考虑所有 READ/WRITE 操作组合，而在推理时只采用其中一种组合。

HMT第 3 节介绍了所提出的隐马尔可夫 Transformer。

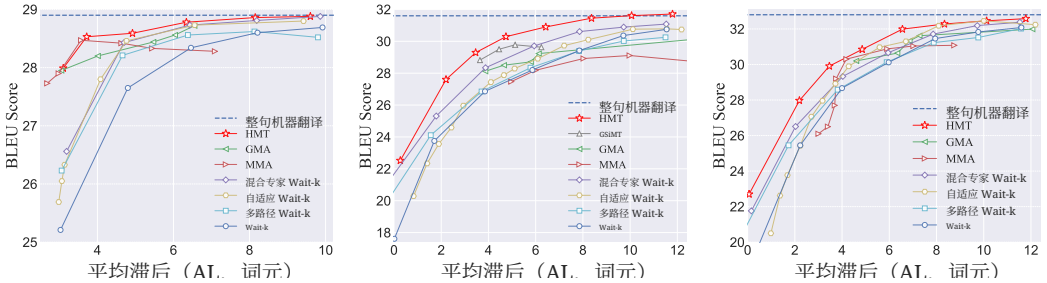
设置所有系统均基于 Fairseq库（Ott 等，2019）中的 Transformer（Vaswani 等，2017）。遵循 Ma 等（2020）和 Miao 等（2021）的方法，对于 En→Vi 使用小型 Transformer（4 个注意力头），对于 De→En 使用 Transformer-Base（8 个注意力头）和 Transformer-Big（16 个注意力头）。由于训练复杂度较高，我们仅报告在 De→En 上使用 Transformer-Base 的 GSiMT 结果，这与其原始设置相同（Miao 等，2021）。超参数设置见附录D。

评估对于同时机器翻译的性能，我们使用 BLEU分数（Papineni 等，2002）评估翻译质量，并使用平均滞后（AL）（Ma 等，2019）评估延迟。AL 衡量输出落后于输入的词元数量。为便于比较，我们调整 HMT 中的 L 和 K，以获得不同延迟下的翻译质量；L 和 K 的具体设置见附录E

4.3 主要结果

我们在图4中将 HMT 与现有的同时机器翻译方法进行了比较，可以看到，HMT 在所有延迟下均优于以往方法。与固定方法 Wait-k、多路径 Wait-k 和混合专家 Wait-k 相比，HMT 具有明显优势，因为它能够动态判断何时开始翻译，从而处理复杂的输入（Arivazhagan 等，2019）。与自适应方法相比，

[4] nlp.stanford.edu/projects/nmt/ [5] statmt.org/wmt15/translation-task.html [6] github.com/ictnlp/MoE-Waitk [7] github.com/facebookresearch/fairseq/tree/main/examples/simultaneous_翻译 [8] github.com/ictnlp/GMA



(a) 英语→越南语，小型 Transformer (b) 德语→英语，Transformer-Base (c) 德语→英语，Transformer-Big

Fi图 4：HMT 与以往同时机器翻译方法的翻译质量（BLEU）与延迟（平均滞后）的对比。

HMT 凭借两大优势，优于当前最先进的 MMA 和 GSiMT。首先，HMT 对开始翻译的时刻进行建模；与 READ/WRITE 操作相比，这些时刻与观测目标序列之间的相关性更强（Zhang & Feng, 2022c）。其次，MMA 和 GSiMT 在推理时都只能考虑 READ/WRITE 动作的一种组合（Ma 等，2020；Miao 等，2021），而 HMT 在训练和推理时都可以考虑多个开始翻译的时刻，因为 HMT 会针对不同的翻译时刻显式生成多个状态。此外，MMA 和 GSiMT 在某些延迟下性能会下降（Ma 等，2020），这主要是因为训练时考虑过多 READ/WRITE 动作组合可能会导致相互干扰（Elbayad 等，2020；Zhang 和 Feng，2021c；Wu 等，2021）。HMT 通过所提出的边界预先剪除不利的开始翻译时刻，从而在所有延迟下取得更稳定、更好的性能。

4.4 训练和推理速度

我们在 De→En 上使用 Transformer-Base，将 HMT 的训练和推理速度与以往方法进行比较，结果如表1所示。所有速度均在 NVIDIA 3090 GPU 上评估。

表1：HMT 的训练和推理速度。

系统		参数数量	训练（秒/批次）	推理（秒/词元）
整句机器翻译		60.9M	0.204	0.0097
Wait-k		60.9M	0.205	0.0108
MMA		62.5M	2.112	0.0647
GSiMT		60.9M	5.090	0.0247
HMT	$L=2, K=4$	60.9M	0.531	0.0204
	$L=5, K=6$	60.9M	0.730	0.0162
	$L=9, K=8$	60.9M	1.051	0.0142

训练速度 与固定方法 Wait-k 相比，HMT 的训练速度稍慢，因为它会将目标序列上采样 K 倍。考虑到性能有明显提升，我们认为，与固定方法相比，训练速度稍慢是完全可以接受的。与自适应方法 MMA 和 GSiMT 相比，后两者会循环计算变量的表示以指示读取/写入（Ma 等，2020；Miao 等，2021），而 HMT 得益于并行计算多个状态的表示，在训练速度上具有明显优势。此外，与考虑所有可能读取/写入组合的 GSiMT 相比，所提出的翻译时刻下界和上界也有效加快了 HMT 训练。

推理速度 与 MMA 相比，HMT 只需少量额外参数（式（5）中的$\mathbf{W}^S$），因而推理速度更快；MMA 则在每个注意力头中增加更多额外参数来预测读取/写入操作（Ma 等，2020）。与 GSiMT 相比，HMT 会预先剪除不理想的翻译时刻，只在其余有价值的时刻中进行判断，从而提升推理速度；而 GSiMT 需要在每一步都计算目标表示并生成读取/写入（Miao 等，2021）。需要注意的是，随着下界 L 增大，HMT 可以剪除更多候选翻译时刻，从而显著加快推理速度。

5 分析

我们进行了广泛的分析，以研究 HMT 的具体改进。除非另有说明，所有结果均在使用 Transformer-Base 的 De→En 测试集上报告。

表 2：HMM 损失的消融研究，包括对所有可能的选择进行边缘化，或仅优化最可能的选择结果。

	AL	BLEU
对所有可能的选择进行边缘化的选择	**3.46**	**29.29**
优化最可能的选择	1.77	24.02

表3：延迟损失权重 $\lambda_{latency}$ 的消融研究 $\mathcal{L}_{latency}$。

$\lambda_{latency}$	AL	BLEU
0.0	6.66	30.07
0.5	4.12	29.80
1.0	**3.46**	**29.29**
1.5	2.10	26.65
2.0	2.00	25.94

表4：状态损失权重 λ_{state} 的消融研究 $\mathcal{L}_{state}$。

λ_{state}	AL	BLEU
0.0	3.24	27.74
0.5	3.47	29.10
1.0	**3.46**	**29.29**
1.5	3.46	29.16
2.0	3.58	29.21

5.1 消融研究

我们进行了多项消融研究，其中 $L=3$ 和 $K=6$ 应用于所有消融研究。

HMM 损失HMT 在训练期间通过对所有可能的选择结果进行边缘化，来学习何时开始翻译。为了验证其有效性，我们在表2中比较了对所有可能的选择进行边缘化与仅优化最可能的选择的性能；后者通过将式（7）中的 $\sum_{\mathbf{z}}$ 替换为 $\text{Max}_{\mathbf{z}}$ 实现。在 $L=3$ 和 $K=6$ 设置相同的情况下，仅优化最可能的选择会使模型陷入局部最优（Miao 等，2021），即始终选择第一个状态，因此其延迟和翻译质量都接近 wait-3策略。对所有可能的选择进行边缘化，能够有效地使 HMT 学会从多个可能时刻中选择开始翻译的时机，从而在翻译质量与延迟之间实现更好的权衡。

延迟损失的权重表3报告了 HMT 在不同 $\lambda_{latency}$下的性能。$\lambda_{latency}$ 的设置会影响延迟与翻译质量之间的权衡。过大（即 2.0）或过小（即 0.0）的 $\lambda_{latency}$ 有时会使模型在到达上界或下界时开始翻译，而适中的 $\lambda_{latency}$ 则能实现最佳权衡。值得注意的是，HMT 对 $\lambda_{latency}$的设置并不敏感，$\lambda_{latency}=0.5$ 或 $\lambda_{latency}=1.0$ 都能实现最佳权衡，其中采用 $\lambda_{latency}=0.5$ 得到的结果几乎落在图4(b)中的 HMT 曲线上。

状态损失的权重表4展示了引入状态损失 $\mathcal{L}_{state}$的有效性。$\mathcal{L}_{state}$ 促使每个状态生成正确的目标词元，从而使 BLEU 提高约 1.5 分。此外，HMT 的翻译质量对状态损失的权重 λ_{state} 并不敏感，不同的 λ_{state} 都能带来相似的提升。

5.2 状态越多越好吗?

HMT 为每个目标词元生成一组 K 状态，以捕捉多个开始翻译的时刻。为了探究多少个状态更好，我们在不同的 K 下报告 HMT 的性能，如表5所示。为进行比较，我们采用不同的下界 L（参见公式（1））， 以获得相近的延迟。

结果表明，考虑多个状态显著优于仅考虑一个状态（$K=1$），这说明 HMT 能从多个状态中找到更好的开始翻译时刻。对于多个状态，较大的状态数量 K 并不一定会带来更好的同时机器翻译性能，而且 HMT 在不同延迟水平下对 K 表现出特定偏好。具体而言，较小的 K 在低延迟下表现更好，而最佳 K 会随着延迟增加而逐渐增大。这是因为间隔较大的翻译时刻可能会在训练过程中相互干扰（Elbayad

表5：不同状态数量下的 HMT 性能 K。

	L	K	AL	BLEU
Low 延迟	4	1	2.57	25.57
	3	2	2.15	26.07
	2	**4**	**2.20**	**27.60**
	1	6	2.28	25.69
中 延迟	7	1	5.86	28.20
	6	4	4.90	30.14
	5	**6**	**4.74**	**30.29**
	4	8	4.69	29.35
High 延迟	11	1	9.71	30.36
	10	6	9.06	31.32
	9	**8**	**8.36**	**31.45**
	8	10	8.27	31.36

等人，2020年；Zhang 等，2021；Zhang 和 Feng，2021c），其中这种差距在低延迟下更为明显。以 $K=6$ 为例，读取 1 个词元/6 个词元（即低延迟）之间的差距比读取 10 个词元/15 个词元（即高延迟）更明显，因为后者在已接收的源词元上包含更多重叠。得益于所提出的翻译时刻边界，HMT 可通过针对不同延迟水平设置合适的 K 来避免这种干扰。此外，与在训练期间直接考虑任意读取/写入组合的 GSiMT（Miao 等，2021）相比，HMT 会预先剪除那些不利的时刻，从而获得更好的性能。

5.3 状态间注意力的优越性

状态间自注意力（见式（2））使 HMT 能够考虑多个开始翻译的时刻，从而捕获全面的状态表示。为验证状态间自注意力的有效性，我们比较了针对每个目标词元关注多个状态与仅关注一个状态的性能。为此，我们采用了三种自注意力模式，分别命名为*Multiple*、*Max* 和选定。“Multiple”是 HMT 中提出的自注意力机制，其中状态可以关注多前几个目标词元的状态。“Max”表示，该状态只能关注每个目标词元中翻译时刻最大的一个状态。“选定”表示，该状态只能关注用于生成每个目标词元的选定状态，这是现有同时机器翻译方法中最常见的推理方式（即，一旦同时机器翻译模型决定开始翻译，后续翻译将关注由这一决定产生的目标表示。）（Ma 等，2019；2020）。请注意，所有注意力模式都需要避免状态之间的信息泄漏，即满足$t_{j,\ k'} \leq t_{i,\ k}$公式（2）中的条件。有关这些注意力的详细介绍，请参见附录 C.6。

表 6：HMT 中不同自注意力模式的性能。

训练	推理	AL	BLEU
Multiple	Multiple	**3.46**	**29.29**
Multiple	Max	3.37	28.99
Multiple	选定	3.42	28.83
Max	Max	3.30	28.66
Max	选定	3.36	28.46

如表 6 所示，在训练和推理中均使用“Multiple”可获得最佳性能。与“Max”相比，后者聚焦于翻译时刻最大的状态（即包含最多源信息的状态），“Multiple”通过考虑多个不同的翻译时刻，使 BLEU 提高了 0.65（Zhang 和 Feng，2021c）。在推理过程中，考虑具有不同翻译时刻的多个状态可以有效提高鲁棒性，因为同时机器翻译模型在接收到一些新的源端词元后，可能会发现自己在前一步做出了错误的决策（Zheng 等，2020b）。对于“Selection”，如果同时机器翻译模型错误地判断了何时开始翻译，后续翻译就会受到影响，因为它们只能聚焦于这个选定的状态。得益于状态间注意力，HMT 使后续翻译能够聚焦于那些未选定的状态，从而减轻不确定决策的影响，并使 BLEU 提高约 0.46。

5.4 选择质量

HMT 通过预测状态 $s_{i,k}$ 的置信度$c_{i,k}$来判断是否选择该状态。为验证基于预测置信度进行选择的质量，我们在图 5 中计算了不同置信度下的词元准确率。置信度与词元准确率之间存在明显的相关性：对于能够更准确地生成目标词元的状态，HMT 会赋予更高的置信度。尤其是在推理过程中，由于 HMT 会在状态 $s_{i,k}$ 的置信度 $c_{i,k} \geq 0.5$时选择该状态来生成目标词元，因此 HMT 学会在平均准确率超过 60% 的时刻开始翻译，从而确保翻译质量。

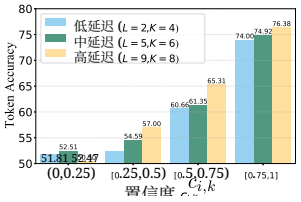


图 5：预测置信度下的词元准确率。

除了词元准确率之外，我们还进一步分析了整个序列上翻译质量与选择结果之间的关系。我们在图6中报告了不同选择结果概率 $p(\mathbf{z})$ 下的 BLEU 分数，并采用 $p(\mathbf{z})^{\frac{1}{|\mathbf{z}|}}$，以避免目标长度 $|\mathbf{z}|$ 对 $p(\mathbf{z})$产生影响。概率较高的选择结果总是能取得更高的 BLEU 分数，这表明，基于边缘似然进行训练会促使 HMT 为能够生成高质量观测目标序列的隐藏序列（即整个序列上的选择结果）赋予更高的概率。

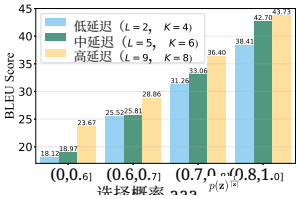


图6：不同选择结果概率下的 BLEU 分数。

6 结论

本文提出了一种用于同时机器翻译的隐马尔可夫 Transformer（HMT），将学习何时开始翻译与学习翻译整合到一个统一框架中。在多个SiMT基准测试上的实验表明，HMT优于强基线模型，并实现了最先进的性能。此外，广泛的分析证明了HMT的有效性和优越性。

致谢

我们感谢所有匿名审稿人提出的深刻且宝贵的意见。

参考文献

Ashkan Alinejad、Hassan S. Shavarani 和 Anoop Sarkar。用于同步神经机器翻译策略生成的基于翻译的监督。载于*2021*年自然语言处理实证方法会议论文集，第 1734–1744 页，线上及多米尼加共和国蓬塔卡纳，2021年11月。计算语言学协会。网址：https: // aclanthology.org/2021.emnlp-main.130。

Naveen Arivazhagan、Colin Cherry、Wolfgang Macherey、Chung-Cheng Chiu、Semih Yavuz、Ruom- ing Pang、Wei Li 和 Colin Raffel。用于同步机器翻译的单调无限回看注意力。载于第*57*届计算语言学协会年会论文集，第 1313–1323 页，意大利佛罗伦萨，2019年7月。计算语言学协会。doi：10.18653/v1/P19-1126。网址：https://aclanthology.org/P19-1126。

Leonard E. Baum 和 Ted Petrie。有限状态马尔可夫链概率函数的统计推断。《数学统计年刊》，37(6):1554–1563，1966年。ISSN 00034851。网址：http://www.jstor.org/stable/2238772。

Kyunghyun Cho 和 Masha Esipova。神经机器翻译能否实现 simultaneous translation？2016年。网址：http://arxiv.org/abs/1606.02012。

Fahim Dalvi、Nadir Durrani、Hassan Sajjad 和 Stephan Vogel。神经机器翻译中 simultaneous translation 的增量解码和训练方法。载于*2018*年计算语言学协会北美分会会议论文集：人类语言技术，第*2*卷（短篇论文），第493–499页，新奥尔良，路易斯安那州，2018年6月。计算语言学协会。doi：10.18653/v1/N18-2079。网址：https://www.aclweb.org/anthology/N18-2079。

Maha Elbayad、Laurent Besacier 和 Jakob Verbeek。《高效的同步机器翻译 Wait-k模型》，2020年。网址：http://dx.doi.org/10.21437/Interspeech. 2020-1241。

Alvin Grissom II、He He、Jordan Boyd-Graber、John Morgan 和 Hal Daum´e III。《别等到最后一个动词：用于同步机器翻译的强化学习》。载于《*2014*年自然语言处理实证方法会议论文集（*EMNLP*）》，第1342–1352页，多哈，卡塔尔，2014年10月。计算语言学协会。doi：10. 3115/v1/D14-1140。网址：https://www.aclweb.org/anthology/D14-1140。

Jiatao Gu、Graham Neubig、Kyunghyun Cho 和 Victor O.K. Li。《利用神经机器翻译学习实时翻译》。载于《欧洲计算语言学协会分会第*15*届会议论文集：第*1*卷，长篇论文》，第1053–1062页，瓦伦西亚，西班牙，2017年4月。计算语言学协会。网址：https: // www.aclweb.org/anthology/E17-1099。

Shoutao Guo、Shaolei Zhang 和 Yang Feng。将固定转为自适应：将后评估融入同步机器翻译。载于 计算语言学协会论文集：*EMNLP 2022*，第 2264–2278 页，阿布扎比，阿拉伯联合酋长国，2022年12月。计算语言学协会。网址：https://aclanthology.org/2022. findings-emnlp.167。

Sathish Reddy Indurthi、Mohd Abbas Zaidi、Beomseok Lee、Nikhil Kumar Lakumarapu 和 Sangha Kim。用于 simultaneous translation 的语言模型增强型单调注意力。载于*2022*年北美计算语言学协会分会会议论文集：人类语言技术，第 38–45 页，美国西雅图，2022年7月。计算语言学协会。doi：10.18653/v1/2022.naacl-main.3。URL https://aclanthology.org/2022.naacl-main.3。

Dan Liu、Mengge Du、Xiaoxi Li、Ya Li 和 Enhong Chen。用于 simultaneous translation 的交叉注意力增强型传导网络。载于*2021*年自然语言处理实证方法会议论文集，第 39–55 页，线上及多米尼加共和国蓬塔卡纳，2021年11月。计算语言学协会。URL https: //aclanthology.org/2021.emnlp-main.4。

Mingbo Ma、Liang Huang、Hao Xiong、Renjie Zheng、Kaibo Liu、Baigong Zheng、Chuanqiang Zhang、Zhongjun He、Hairong Liu、Xing Li、Hua Wu 和 Haifeng Wang。STACL：使用前缀到前缀框架，通过隐式预判和可控延迟实现 simultaneous translation。载于第*57*届计算语言学协会年会论文集，第3025–3036页，意大利佛罗伦萨，2019年7月。计算语言学协会。doi：10.18653/ v1/P19-1289。URL https://www.aclweb.org/anthology/P19-1289。

Xutai Ma、Juan Miguel Pino、James Cross、Liezl Puzon 和 Jiatao Gu。单调多头注意力。载于国际学习表征会议，2020年。URL https:// openreview.net/forum?id=Hyg96gBKPS。

Yishu Miao、Phil Blunsom 和 Lucia Specia。《同步机器翻译的生成式框架》。载于*2021*年自然语言处理实证方法会议论文集，第 6697–6706 页，线上及多米尼加共和国蓬塔卡纳，2021年11月。计算语言学协会。网址：https://aclanthology.org/2021.emnlp-main.536。

Myle Ott、Sergey Edunov、Alexei Baevski、Angela Fan、Sam Gross、Nathan Ng、David Grangier 和 Michael Auli。《fairseq：用于序列建模的快速、可扩展工具包》。载于*2019*年计算语言学协会北美分会年会（演示）论文集，第 48–53 页，明尼阿波利斯，明尼苏达州，2019年6月。计算语言学协会。doi：10.18653/v1/N19-4009。网址：https://www.aclweb.org/anthology/N19-4009。

Kishore Papineni、Salim Roukos、Todd Ward 和 Wei-Jing Zhu。《BLEU：一种机器翻译自动评估方法》。载于计算语言学协会第*40*届年会论文集，第311–318页，美国宾夕法尼亚州费城，2002年7月。计算语言学协会。doi: 10.3115/1073083.1073135。URL https: //www.aclweb.org/anthology/P02-1040。

L. Rabiner 和 B. Juang。《隐马尔可夫模型导论》。《style id='1'>IEEE ASSP Magazine，3(1):4–16，1986年。doi: 10.1109/MASSP.1986.1165342。URL https://ieeexplore.ieee.org/ document/1165342。

Colin Raffel、Minh-Thang Luong、Peter J. Liu、Ron J. Weiss 和 Douglas Eck。通过强制单调对齐实现线上线性时间注意力。载于 Doina Precup 和 Yee Whye Teh（编），第*34*届国际机器学习会议论文集，第70卷，机器学习研究论文集，第2837–2846页。PMLR，2017年8月6–11日。URL https://proceedings.mlr.press/v70/raffel17a.html。

Rico Sennrich、Barry Haddow 和 Alexandra Birch。利用子词单元对稀有词进行神经机器翻译。载于第*54*届计算语言学协会年会论文集（第*1*卷：长篇论文），第1715–1725页，德国柏林，2016年8月。计算语言学协会。doi：10.18653/v1/P16-1162。网址：https://www.aclweb.org/anthology/P16-1162。

Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Ł ukasz Kaiser 和 Illia Polosukhin。注意力即你所需的一切。载于 I. Guyon、U. V. Luxburg、S. Bengio、H. Wallach、R. Fergus、S. Vishwanathan 和 R. Garnett（编），神经信息处理系统进展 *30*，第5998–6008页。Curran Associates, Inc.，2017年。网址：http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf。

Weiyue Wang、Derui Zhu、Tamer Alkhouli、Zixuan Gan 和 Hermann Ney。用于机器翻译的神经隐马尔可夫模型。载于计算语言学协会第*56*届年会论文集（第*2*卷：短文），第377–382页，澳大利亚墨尔本，2018年7月。计算语言学协会。doi：10.18653/v1/P18-2060。URL https: //aclanthology.org/P18-2060。

Xueqing Wu、Lewen Wang、Yingce Xia、Weiqing Liu、Lijun Wu、Shufang Xie、Tao Qin 和 Tie-Yan Liu。用于序列学习的时间相关任务调度。载于 MarinaMeila 和 Tong Zhang（编），第 *38*届国际机器学习会议论文集，《机器学习研究论文集》第139卷，第11274–11284页。PMLR，2021年7月18日至24日。URL https://proceedings.mlr.press/v139/wu21e.html。

Shaolei Zhang 和 Yang Feng。ICT 的 AutoSimTrans 2021 系统：稳健的字符级 simultaneous translation。载于第二届自动同声翻译研讨会论文集，第 1–11 页，线上，2021年6月a。计算语言学协会。doi：10.18653/v1/2021.autosimtrans-1.1。URL：https://aclanthology.org/2021.autosimtrans-1.1。

Shaolei Zhang 和 Yang Feng。使用高斯混合模型对神经机器翻译中的集中式交叉注意力进行建模。载于计算语言学协会论文集：*EMNLP 2021*，第 1401–1411 页，多米尼加共和国蓬塔卡纳，2021年11月b。计算语言学协会。doi：10.18653/v1/2021.findings-emnlp.121。URL：https://aclanthology.org/2021.findings-emnlp.121。

Shaolei Zhang 和 Yang Feng。采用混合专家 wait-k 策略的通用同步机器翻译。载于 *2021*年自然语言处理实证方法会议论文集，第 7306–7317 页，线上及多米尼加共和国蓬塔卡纳，2021年11月c。计算语言学协会。doi：10.18653/v1/2021.emnlp-main.581。URL：https://aclanthology.org/2021.emnlp-main.581。

Shaolei Zhang 和 Yang Feng。为同步机器翻译建模双重读写路径。载于计算语言学协会第 *60* 届年会论文集（第 *1* 卷：长篇论文），第 2461–2477 页，爱尔兰都柏林，2022年5月a。计算语言学协会。URL：https://aclanthology.org/2022.acl-long.176。

Shaolei Zhang 和 Yang Feng。采用长度感知框架减少同步机器翻译中的位置偏差。载于计算语言学协会第 *60* 届年会论文集（第 *1* 卷：长篇论文），第 6775–6788 页，爱尔兰都柏林，2022年5月b。计算语言学协会。URL：https://aclanthology.org/2022.acl-long.467。

Shaolei Zhang 和 Yang Feng。用于 simultaneous machine translation 的高斯多头注意力。载于计算语言学协会 *ACL 2022* 论文集，第 3019–3030 页，爱尔兰都柏林，2022年5月c。计算语言学协会。doi：10.18653/v1/2022.findings-acl.238。URL：https://aclanthology.org/2022.findings-acl.238。

Shaolei Zhang 和 Yang Feng。用于 simultaneous translation 的基于信息传输的策略。载于 *2022*年自然语言处理实证方法会议论文集，第 992–1013 页，阿拉伯联合酋长国阿布扎比，2022年12月d。计算语言学协会。URL：https://aclanthology.org/2022.emnlp-main.65。

Shaolei Zhang、YangFeng 和 Liangyou Li。用于 simultaneous translation 的未来引导增量式 Transformer。*AAAI* 人工智能会议论文集，35（16）：14428–14436，2021年5月。URL：https://ojs.aaai.org/index.php/AAAI/article/view/17696。

Shaolei Zhang、Shoutao Guo 和 Yang Feng。等待信息策略：在信息层面平衡源与目标的同步机器翻译。载于 计算语言学协会计算语言学研究发现：*EMNLP 2022*，第 2249–2263 页，阿布扎比，阿拉伯联合酋长国，2022 年 12 月。计算语言学协会。网址：https://aclanthology. org/2022.findings-emnlp.166。

Baigong Zheng、Renjie Zheng、Mingbo Ma 和 Liang Huang。通过受限模仿学习，采用灵活策略进行 simultaneous translation。在计算语言学协会第*57*届年会论文集中，第5816–5822页，佛罗伦萨，意大利，2019年7月a。计算语言学协会。doi：10.18653/v1/P19-1582。URL：https://www.aclweb.org/anthology/P19-1582。

Baigong Zheng、Renjie Zheng、Mingbo Ma 和 Liang Huang。为 simultaneous translation 学习自适应策略：更简单、更快速的方法。在*2019*年自然语言处理经验方法会议暨第九届自然语言处理国际联合会议（*EMNLP-IJCNLP*）论文集中，第1349–1354页，中国香港，2019年11月b。计算语言学协会。doi：10.18653/v1/D19-1137。URL：https://www.aclweb.org/anthology/D19-1137。

Baigong Zheng、Kaibo Liu、Renjie Zheng、Mingbo Ma、Hairong Liu 和 Liang Huang。《同步翻译策略：从固定到自适应》。载于计算语言学协会第*58*届年会论文集，第 2847–2853 页，线上，2020年7月a。计算语言学协会。doi: 10.18653/v1/2020.acl-main.254。URL https: //www.aclweb.org/anthology/2020.acl-main.254。

Renjie Zheng、Mingbo Ma、Baigong Zheng、Kaibo Liu 和 Liang Huang。《面向 simultaneous translation 的及时纠正机会式解码》。载于计算语言学协会第*58*届年会论文集，第437–442页，线上，2020年7月b。计算语言学协会。doi: 10.18653/v1/2020.acl-main.42。URL https://aclanthology.org/2020.acl-main.42。

转移概率的计算

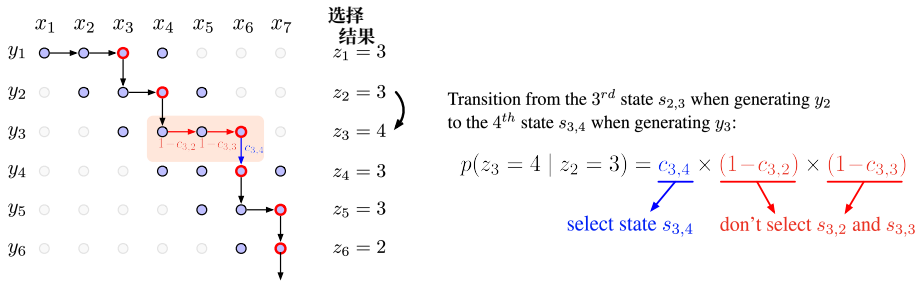


图 7：展示转移概率 $p(z_i \mid z_{i-1})$计算方法的示例。

我们在第3.2节给出了选择结果之间的转移概率 $p(z_i \mid z_{i-1})$，并将在此更详细地介绍转移概率的计算方法。

首先，特定的选择 z_i 表示在通过状态s_i、z_i接收前 y_i t_i、z_i 个源词元时进行翻译。同样，选择 z_{i-1} 表示在通过状态s_{i-1}、z_{i-1}接收前 y_{i-1} t_{i-1}、z_{i-1} 个源词元时进行翻译。由于同时机器翻译中开始翻译时刻的单调性，当t_{i-1}、z_{i-1} $> t_i$、z_i时，不可能从 z_{i-1} 转移到 z_i 。然后，对于t_{i-1}、z_{i-1} $\leq t_i$、z_i，由于 HMT 会逐一判断是否选择每个状态（从s_i、1 到s_i、κ），并在选定某个状态后开始翻译，因此从 z_{i-1} 转移到 z_i 的前提是，翻译时刻属于$[t_{i-1,z_{i-1}}, t_{i,z_i})$的所有状态都不确信会被选定，而$s_i$、$z_i$ 确信会被选定。如图 7 所示，从 z_2 到 z_3 的转移概率$p(z_3 = 4 \mid z_2 = 3)$由不选定s_3、2 和s_3、3 的概率，以及选定s_3、4 的概率组成。

形式上，转移概率 $p(z_i \mid z_{i-1})$计算如下：

$$p(z_i \mid z_{i-1}) = \begin{cases} c_{i,z_i} \times \prod\limits_{t_{i-1,z_{i-1}} \leq t_{i,l} < t_{i,z_i}} (1-c_{i,l}) & \text{if } t_{i,z_i} \geq t_{i-1,z_{i-1}} \\ 0 & \text{if } t_{i,z_i} < t_{i-1,z_{i-1}} \end{cases} . \tag{12}$$

B HMT 中的动态规划

HMT 将选择哪些状态（即何时开始翻译）视为隐事件，并将目标词元视为观测事件，同时将目标序列的生成组织为隐马尔可夫模型。在训练过程中，HMT 通过最大化所有可能的选择结果（即隐事件）上的目标序列（即观测事件）的边缘似然来进行优化：

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}) \times p(\mathbf{z}) \tag{13}$$

$$= \sum_{\mathbf{z}} \prod_{i=1}^{|\mathbf{y}|} p\left(y_i \mid \mathbf{x}_{\leq t_{i,z_i}}, \mathbf{y}_{<i}, z_i\right) \times p(z_i \mid z_{i-1}) . \tag{14}$$

我们应用动态规划（即 HMM 中的前向算法）来高效计算边缘似然（Baum 和 Petrie, 1966）。形式上，我们引入中间变量 $\alpha_i(k)$，表示生成前 i 个目标词元 $\mathbf{y}_{\leq i}$ 时选择 k^{th} 状态 $s_{i,k}$ 的概率，其定义如下：

$$\alpha_i(k) = p(\mathbf{y}_{\leq i}, z_i = k \mid \mathbf{x}) . \tag{15}$$

初始化 初始 $\alpha_1(k)$计算如下：

$$\alpha_1(k) = \pi_k \times p\left(y_1 \mid \mathbf{x}_{\leq t_{1,k}}, \mathbf{y}_{<1}, z_1 = k\right) , \tag{16}$$

其中，$\pi_k = p(z_1 = k)$ 是选择 $s_{1,k}$ 的初始概率。在实现中，我们在 z_1 之前添加一个特定的选择 z_0，其值为 $p(z_0) = 1$，表示尚未收到任何源词元。

在翻译开始时，即 $t_{0,z_0}=0$。因此，初始概率 $p(z_1)$ 是通过从 z_0 到 z_1 的转移概率计算得出的。

递归 接下来，$\alpha_i(k)$ 通过对从上一步的选择到当前选择的所有转移扩展求和来计算。因此，$\alpha_i(k)$ 按以下递归形式计算：

$$\alpha_i(k)=\sum_{k'=1}^{K}\alpha_{i-1}\left(k'\right)\times p\left(z_i=k\mid z_{i-1}=k'\right)\times p\left(y_i\mid \mathbf{x}_{\leq t_{i,k}},\mathbf{y}_{<i},z_i=k\right). \tag{17}$$

终止 最后，对所有可能的选择结果求和，计算目标序列的边缘似然：

$$p(\mathbf{y}\mid\mathbf{x})=\sum_{k=1}^{K}\alpha_I(k). \tag{18}$$

C 扩展分析

C.1 策略与翻译的具体改进

所提出的 HMT 将同时机器翻译中的两个关键问题——“学习何时开始翻译”（即推理策略）和“学习翻译”（即翻译能力）——整合到一个统一框架中。由于 HMT 在训练和推理阶段都对多个开始翻译的时刻进行显式建模，HMT 架构能够灵活地与其他推理策略（例如 wait-k 策略（Ma 等，2019））配合使用，从而使我们能够了解 HMT 架构和推理策略各自带来的具体改进。具体而言，我们在图 8 中报告了“HMT 架构 + Wait-k 推理”的结果，其中 HMT 架构的 wait-k 推理通过强制 HMT 始终选择最后一个状态来实现。

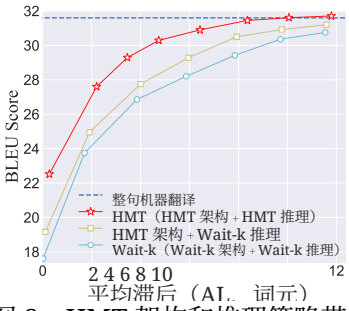


图 8：HMT 架构和推理策略带来的具体改进。

相比之下，对于 wait-k 推理，HMT 架构凭借全面的.....考虑多个翻译时刻，从而提升了 0.8 BLEU。此外，与 Wait-k 推理相比，HMT 推理能够学习更准确的开始翻译时刻，平均带来 2.8 BLEU 的提升。尤其是在低延迟场景下，推理策略带来的提升更为明显，因为在低延迟的同时机器翻译中，准确的翻译时刻更为重要（Arivazhagan 等，2019）。

C.2 置信度阈值的消融研究

在推理过程中，当置信度 $c_{i,k}\geq 0.5$ 时，HMT 会选择状态 $s_{i,k}$ 来生成 y_i；其中，0.5 可视为置信度阈值，表示 HMT 认为该状态能够生成正确目标词元。图 5 也证明，置信度较高的状态能够更准确地生成目标词元。为研究置信度阈值对 HMT 的影响，我们在图 9 中展示了 HMT 在不同置信度阈值 δ 下的性能；在推理过程中，当置信度为 $c_{i,k}\geq\delta$ 时，HMT 会选择状态 $s_{i,k}$ 来生成 y_i。

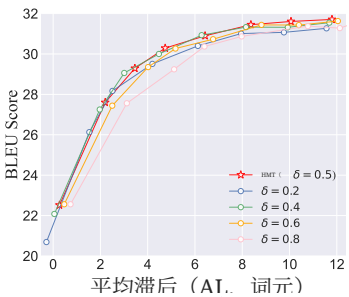


图 9：HMT 在不同置信度阈值 δ 下的性能。

在推理过程中，适中的置信度阈值（如 $\delta=0.4$ 和 $\delta=0.5$）能够达到相近的同时机器翻译性能，这表明 HMT 对置信度设置并不敏感阈值。此外，随着置信度阈值降至 0.2，HMT 开始翻译的时间大幅提前，导致翻译质量略有下降。随着阈值升至 0.8，HMT 的延迟增加，但翻译质量并未明显提升，这表明 0.5 的置信度已足以为 HMT 中的状态生成正确的目标词元。

C.3 案例研究

我们针对图 10 和图 11 中两个较难的德语→英语案例开展了案例研究。这些案例中，德语与英语之间的语序差异给同时机器翻译模型带来了更大的挑战（即德语中的动词总是滞后出现）（Ma 等，2019；Zhang 和 Feng，2021c）。我们展示了 HMT 的具体推理过程，并将所有状态（包括选定、未选定和未考虑的状态）的输出可视化，以说明 HMT 能够成功学会何时开始翻译。

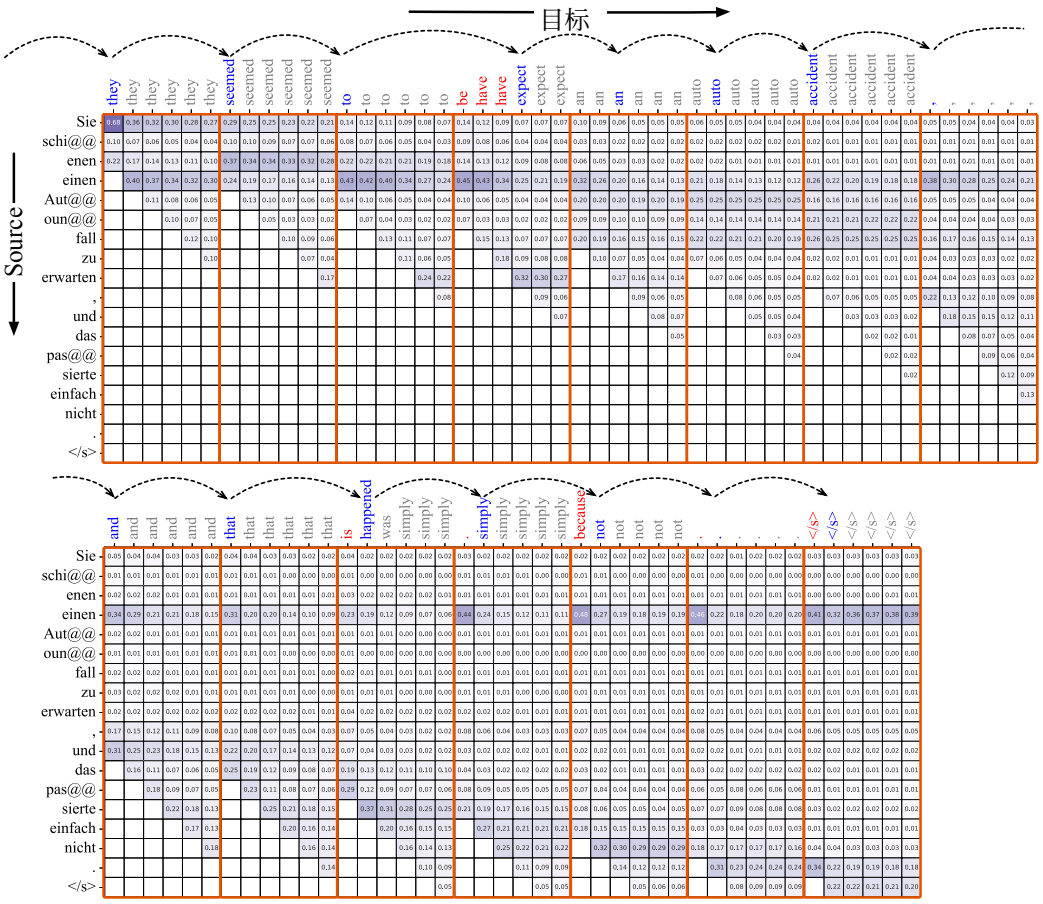
| | | |

见图 5。此外，选定的状态始终倾向于能够生成正确翻译的最早状态，即延迟更低的状态。

源：	他们似乎在等一场车祸发生，但事情就是没有发生。 </s>
	他们 似乎 a 车祸 期待 而那 发生 _就这样 _没有发生 _._ </s>
参考：	他们似乎预料会发生一场车祸，但事情并没有发生。</s>

Step	输入（接收到的源序列）	状态	置信度	输出
1	他们似乎	$s_{1,1}$	0.51	they
2	他们似乎有一个	$s_{2,1}$	0.54	似乎
3	他们似乎有一辆汽车	$s_{3,1}$	0.87	to
4	他们似乎发生了汽车事故	$s_{4,1}$	0.09	~~be~~
5	他们似乎发生了汽车事故	$s_{4,2}$	0.14	~~have~~
6	他们似乎发生了汽车事故	$s_{4,3}$	0.07	~~have~~
7	他们似乎发生了汽车事故	$s_{4,4}$	0.88	预期
8	他们似乎发生了汽车事故	$s_{5,3}$	0.91	an
9	他们似乎预料到一场车祸	$s_{6,2}$	0.94	auto
10	他们似乎预料到一场车祸	$s_{7,1}$	0.67	事故
11	他们似乎预料到一场车祸，	$s_{8,1}$	0.90	,
12	他们似乎预料到一场车祸，并且	$s_{9,1}$	0.64	and
13	他们似乎预料到一场车祸，并且这	$s_{10,1}$	0.55	that
14	他们似乎预料到会发生一场车祸，而这	$s_{11,1}$	0.24	~~is~~
15	他们似乎预料到会发生一场车祸，而这发生了	$s_{11,2}$	0.54	发生了
16	他们似乎预料到会发生一场车祸，而这发生了	$s_{12,1}$	0.06	~~,~~
17	他们似乎预料到会发生一场车祸，而这就这么发生了	$s_{12,2}$	0.56	就这么
18	他们似乎预料会发生车祸，而事情就这样发生了	$s_{13,\ 1}$	0.03	~~因为~~
19	他们似乎预料会发生车祸，而事情就这样没有发生	$s_{13,\ 2}$	0.97	not
20	他们似乎预料会发生车祸，而事情就这样没有发生	$s_{14,\ 1}$	0.17	~~,~~
21	他们似乎预料会发生车祸，而事情就这样没有发生。	$s_{14,\ 2}$	0.95	.
22	他们似乎预料会发生车祸，而事情就这样没有发生。	$s_{15,\ 1}$	0.40	~~</s>~~
23	他们似乎预料会发生一场车祸，但事情并没有发生。 </s>	$s_{15,\ 2}$	1.00	</s>

(a) HMT 的推理过程。

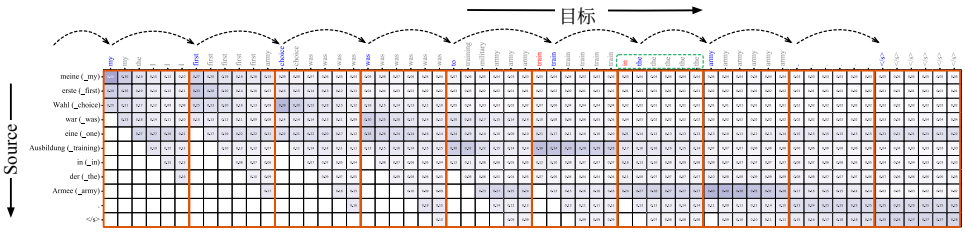


(b) 所有状态输出及其对已接收源词元的交叉注意力可视化。

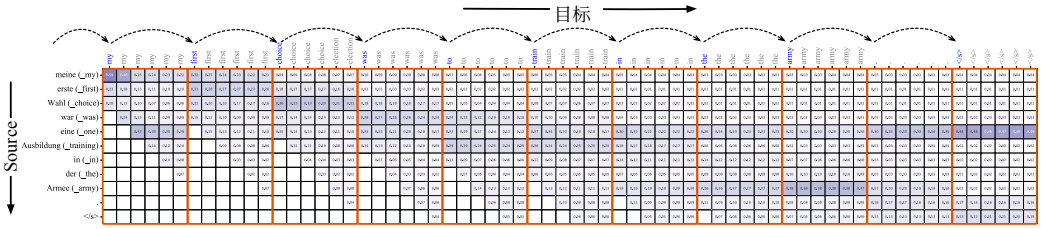
图 11：De→En 测试集中的 #378 案例研究，其中我们使用 HMT，并采用 $L=3$ 和 $K=6$.

动词滞后的情况 图 11 展示了一个更复杂的情况：德语中的动词“*erwarten*”和“*passierte*”出现滞后，这要求同时机器翻译模型准确判断何时开始翻译（Grissom II 等人，2014；Ma 等，2019）。在 HMT 中，当尚未接收到动词时，各状态的置信度始终较低，因此 HMT 可以在接收到源动词后开始翻译相应的“*expect*”和“*happened*”。此外，图 11(b) 中的交叉注意力表明，HMT 在状态 $s_{4,\,4}$ 和 $s_{11,\,2}$ 生成了正确的翻译，因为这些状态关注了已接收的动词“*erwarten*”和“*passierte*”（Zhang 和 Feng, 2021b），证明了根据目标表示和已接收的源内容预测置信度的有效性。

C.4 状态损失的影响 $\mathcal{L}_{state}$



(a) 不使用状态损失 $\mathcal{L}_{state}$的 HMT。翻译结果：‘我最初的选择是训练军队。’。gree n box 标记了错误的翻译，正确的翻译应为‘在’，但模型生成了‘这 ’.



（b）使用状态损失 $\mathcal{L}_{state}$的 HMT。翻译结果：‘我最初的选择是在军队中接受训练。’。

图 12：所提出的状态损失的效果 $\mathcal{L}_{state}$。我们使用 $L = 3$ 和 $K = 6$ 应用 HMT，并将 De→En 测试集中的案例#912的状态输出可视化（源：‘我的第一选择是在军队中接受训练。’；参考译文：‘我最初的选择是参军。’）。红色标记的输出表示未选定的状态，蓝色标记的输出表示选定的状态，灰色标记的输出表示推理时未考虑的状态。方框中的数值表示交叉注意力权重，空白方框表示翻译目标词元时尚未接收到相应的源词元。

对于 HMT训练（参见第3.2节），我们提出状态损失 $\mathcal{L}_{state}$，以鼓励 HMT 无论选定哪个状态（即无论何时开始翻译），都生成正确的目标词元；第5.1节中的消融研究展示了状态损失带来的改进。为进一步研究状态损失 $\mathcal{L}_{state}$的影响，我们在图 12 中可视化了案例#912在有和没有 $\mathcal{L}_{state}$ 时的状态输出。

在训练中移除状态损失 $\mathcal{L}_{state}$ 后，低概率选中的状态几乎无法学会生成正确的目标词元，因为这些状态的发射概率可能对目标序列的边缘似然贡献甚微。如图 12(a) 所示，在生成第一个目标词元时，一些较晚的状态错误地生成了“*I*”；在生成第三个目标词元时，一些状态生成了“*was*”，等等。虽然在大多数情况下，HMT 在推理时不会选择这些状态，但当选择稍有不确定时，模型可能会输出错误的目标词元，例如生成“*the*”而非“*in*”（图 12(a) 中绿色方框所示）。如图 12(b) 所示，加入状态损失 $\mathcal{L}_{state}$ 后，HMT 中的多个状态都能学会生成正确的目标词元，无论何时开始翻译该目标词元。因此，即使 HMT 在选择时做出了错误决策，仍然能够生成正确的目标词元，从而提高选择的鲁棒性。

C.5 策略质量

策略质量直接影响同时机器翻译（SiMT）的性能。良好的策略应确保模型在翻译每个目标词元之前接收到与之对应的源词元，从而实现高质量翻译。按照 Zhang & Feng (2022a) 的做法，我们计算翻译前已接收到的真实对齐源端词元所占比例，以评估策略质量。我们使用 RWTH[9] De→En 对齐数据集，并通过强制解码 [10] 获取每个目标词元的翻译时刻。具体而言，我们将 y_i 的真实对齐源位置[11] 记为 a_i，并用 g_i 记录 y_i 的翻译时刻（即翻译 y_i 时已接收的源词元数量）。给定对齐 a_i 和翻译时刻 g_i，翻译前已接收的对齐源词元所占比例计算如下：

$$\text{Proportion} = \frac{1}{|\mathbf{y}|}\sum_{i=1}^{|\mathbf{y}|}\mathbb{1}_{a_i \le g_i}, \tag{19}$$

$$\text{where}\quad \mathbb{1}_{a_i \le g_i} = \begin{cases} 1, & a_i \le g_i \\ 0, & a_i > g_i \end{cases}. \tag{20}$$

评估结果如图 13 所示。与先前的策略相比，在相同延迟下，HMT 在翻译前接收到更多对齐的源词元，这表明 HMT 能够更精准地决定何时开始翻译。得益于策略的优势，在相同延迟下，HMT 能够接收更多对齐的源词元，从而比先前的方法实现更高的翻译质量。

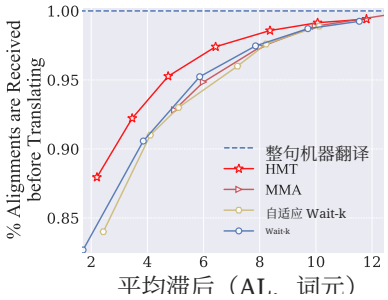
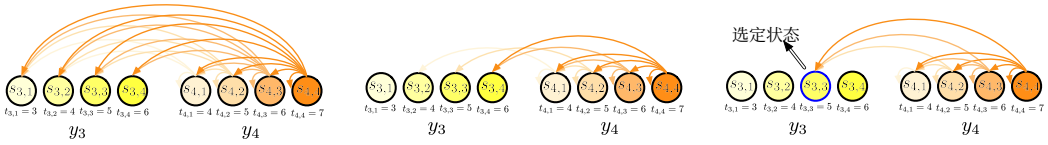


图 13：HMT 中策略的质量。我们计算了各种策略在翻译前接收到的对齐源词元的比例。

C.6 为什么要在状态间使用自注意力?



(a) Multiple　(b) Max　(c) 选定

图14：HMT 中状态间自注意力的示意图，称为“Multiple”。我们还提出了两种变体——“Max”和“Selected”，以展示“Multiple”注意力模式的优越性。该示意图展示了一个包含 $L = 1$ 和 $K = 4$的 HMT 示例，其中 y_3 和 y_4 的状态翻译时刻分别为 $\mathbf{t}_3 = (3,4,5,6)$ 和 $\mathbf{t}_4 = (4,5,6,7)$。

HMT 根据公式 (2) 在所有状态之间应用自注意力，下面我们将更深入地解释 HMT 为何采用这种注意力模式。为便于比较，我们在第5.3节介绍三种自注意力模式，分别称为 Multiple、Max 和 Selected：

- **Multiple**：HMT 中状态间的自注意力模式。在“Multiple”模式下，状态可以关注先前目标词元的多个状态。以图14(a)为例，在翻译 y_4时，状态 $s_{4,3}$ 可以关注 $s_{3,1}$, $s_{3,2}$, $s_{3,3}$ 和 $s_{3,4}$ 中的 y_3；与此同时，状态 $s_{4,3}$ 也会关注 $s_{4,1}$, $s_{4,2}$, $s_{4,3}$ 中的 y_4。
- **Max**：在“Max”中，状态可以关注每个目标词元的一个状态，该状态具有最大的翻译时刻。以图14(b)为例，在翻译 y_4 时，状态 $s_{4,3}$ 只能关注 y_3 的 $s_{3,4}$，因为 $t_{3,4} = 6$ 是具有最大

[9]https://www-i6.informatik.rwth-aachen.de/goldAlignment/[10]强制解码：我们强制模型生成真实目标词元，从而将翻译时刻与真实对齐进行比较。[11]对于多对一对齐，我们选择对齐中的最后一个源位置。

翻译时刻的状态，$s_{4,\ 3}$（$t_{4,\ 3}=6$）可以关注该状态。同时，状态 $s_{4,\ 3}$ 也关注 y_4 的 $s_{4,\ 1}$、$s_{4,\ 2}$、$s_{4,\ 3}$。

- **选定**：现有同时机器翻译方法中最常见的注意力模式。在“选定”模式下，状态可以关注用于生成上一个目标词元的选定状态。一旦同时机器翻译模型决定执行 WRITE（即选定一个状态），后续翻译便只依赖于这一目标表示。以图 14(c) 为例，假设已选定$s_{3,\ 3}$ 来生成 y_3，那么后续的$s_{4,\ 2}$、$s_{4,\ 3}$ 和 $s_{4,\ 4}$ 只能关注$s_{3,\ 3}$的表示。注意，$s_{4,\ 1}$ 不能关注 $s_{3,\ 3}$，因为$t_{3,\ 3} > t_{4,\ 1}$。

Multiple 与 Max表 6 中的结果显示，与“Max”相比，“Multiple”带来了 0.65 BLEU 的提升。“Multiple”和“Max”能够关注的最大翻译时刻相同（在“Multiple”和“Max”中，状态 $s_{4,\ 3}$ 都可以关注 $s_{4,\ 3}$，其中$t_{3,\ 4}=6$。），但“Multiple”还可以考虑那些更早开始翻译的状态，例如 $s_{3,\ 1}$、$s_{3,\ 2}$和 $s_{3,\ 3}$。综合考虑多个状态的表示，有助于“Multiple”做出更精确的判断，并获得更准确的状态表示（Zhang 和 Feng，2021c）。此外，在“Max”中，当前状态总是关注具有最大翻译时刻的一个先前状态，无论哪个状态被选来生成上一个目标词元。例如，即使选定了 $s_{4,\ 2}$ 来生成 y_3，“Max”中的状态 $s_{4,\ 3}$ 仍然只关注 $s_{4,\ 3}$；忽略先前选定的状态 $s_{4,\ 2}$ 会扰乱 y_4 对 y_3 的依赖，从而影响翻译质量。因此，由于能够更全面地关注多个状态，所提出的“Multiple”注意力模式取得了更好的性能。

多个与选定“选定”是当前同时机器翻译方法中最常用的注意力模式，即一旦确定执行写入动作，后续目标词元就只能关注该状态的表示（Ma 等，2019；Arivazhagan 等，2019；Ma 等，2020）。采用“选定”注意力模式的原因在于，以往的方法在推理时只能为每个目标词元保留唯一的翻译时刻及其对应表示，而 HMT 能够显式地为每个目标词元建模多个翻译时刻。仅保留选定翻译时刻的目标表示，容易受到决策不准确的影响（Zheng 等，2020b）。假设模型选择状态 $s_{3,\ 3}$ 来生成 y_3，但这一决策未必完全准确，那么使用 $s_{3,\ 2}$ 或 $s_{3,\ 4}$ 来生成 y_3可能更合理。“选定”要求后续状态只能关注 $s_{3,\ 3}$的表示，这可能导致翻译错误。“多个”则允许后续状态关注多个状态，包括未被选定的状态，从而做出更全面的决策，取得更好的结果。

总之，“Max”忽略先前选定的状态，“Selected”只考虑选定的状态，而提出的“Multiple”注意力模式会关注所有先前状态，并使训练与推理保持一致。因此，在这三种注意力模式中，“Multiple”的表现最佳。

D 超参数

表 7 给出了 HMT 的超参数设置。

E 更多指标下的数值结果

E.1 延迟指标

除了平均滞后（AL）（Ma 等，2019）外，我们还使用连续等待（CW）（Gu 等人，2017）、平均比例（AP）（Cho & Esipova，2016）和可微平均滞后（DAL）（Arivazhagan 等，2019）来评估 HMT 的延迟。我们将翻译 y_i 之前等待的源词元数量记为 g_i（即开始翻译 y_i 的时刻），这些延迟指标的计算方法如下。

连续等待（CW）（Gu 等人，2017）评估两个目标词元之间等待的源词元平均数量。给定 g_i，CW 的计算公式为：

$$\mathrm{CW}=\frac{\sum_{i=1}^{|\boldsymbol{y}|}(g_i-g_{i-1})}{\sum_{i=1}^{|\boldsymbol{y}|}\mathbb{1}_{g_i-g_{i-1}>0}}, \tag{21}$$

表 7：HMT 的超参数。

超参数	En→Vi 小型 Transformer	De→En Transformer-Base	De→En Transformer-Big
编码器层数	6	6	6
编码器注意力头数	4	8	16
编码器嵌入维度	512	512	1024
编码器前馈网络嵌入维度	1024	2048	4096
解码器层	6	6	6
解码器注意力头	4	8	16
解码器嵌入维度	512	512	1024
解码器前馈网络嵌入维度	1024	2048	4096
丢弃	0.3	0.3	0.3
优化器 Adam-β	adam (0.9, 0.98)	adam (0.9, 0.98)	adam (0.9, 0.98)
梯度裁剪范数	0	0	0
lr	2e-4	5e-4	5e-4
学习率调度器	逆平方根	逆平方根	逆平方根
预热更新步数	4000	4000	4000
预热初始学习率	1e-7	1e-7	1e-7
权重衰减	0.0001	0.0001	0.0001
标签平滑	0.1	0.1	0.1
最大令牌数	16000	8192×4	8192×4

其中， $\mathbb{1}_{g_i - g_{i-1} > 0}$ 统计 $g_i - g_{i-1} > 0$的数量。

平均比例（AP）（Cho & Esipova，2016年）用于评估已接收源词元数量与源词元总数之间的比例。给定 g_i，AP 的计算方式如下：

$$\mathrm{AP} = \frac{1}{|\mathbf{x}|\,|\mathbf{y}|} \sum_{i=1}^{|\mathbf{y}|} g_i. \tag{22}$$

平均滞后（AL）（Ma 等，2019）用于评估目标输出相对于源输入的平均滞后词元数。给定 g_i，AL 的计算方式如下：

$$\mathrm{AL} = \frac{1}{\tau} \sum_{i=1}^{\tau} g_i - \frac{i-1}{|\mathbf{y}| \,/\, |\mathbf{x}|}, \quad \text{where } \tau = \operatorname*{argmin}_i \left(g_i = |\mathbf{x}| \right). \tag{23}$$

可微平均滞后（DAL）（Arivazhagan 等，2019）是平均滞后的可微版本。给定 g_i，DAL 的计算方式如下：

$$\mathrm{DAL} = \frac{1}{|\mathbf{y}|} \sum_{i=1}^{|\mathbf{y}|} g_i^{'} - \frac{i-1}{|\mathbf{x}| \,/\, |\mathbf{y}|}, \quad \text{where } g_i^{'} = \begin{cases} g_i & i = 1 \\ \max\left(g_i, g_{i-1}^{'} + \frac{|\mathbf{x}|}{|\mathbf{y}|} \right) & i > 1 \end{cases}. \tag{24}$$

E.2 数值结果

我们调整 HMT 中的 L 和 K（参见公式(1)），以获得不同延迟下的翻译质量。为清晰起见，我们在表 8、表 9 和表 10 中展示了超参数采用特定设置 L 和K时 HMT 的数值结果。请注意，为便于比较，我们设置 $L=-1$，以获得 HMT 在 De→En 上极低延迟条件下的翻译质量（平均滞后 <1，即输出平均比输入滞后不到 1 个词元）。当设置 $L=-1$时，我们将所有状态的翻译时刻 $t_{i,\,k}$限制为至少为 1，即 $t_{i,\,k} = \max\{\min\{L + (i-1) + (k-1) \quad |\mathbf{x}|\},\ 1\}$。因此，所有状态都将在接收到至少 1 个源端词元后开始翻译。

表 8：使用小型 Transformer 的 HMT 在 En→Vi 上的数值结果。

IWSLT15 En→Vi				小型 ***Transformer***		
L	K	**CW**	**AP**	**平均滞后 DAL**		**BLEU**
1	2	1.15	0.64	3.10	3.72	27.99
2	2	1.22	0.67	3.72	4.38	28.53
4	2	1.24	0.72	4.92	5.63	28.59
5	4	1.53	0.76	6.34	6.86	28.78
6	4	1.96	0.83	8.15	8.71	28.86
7	6	2.24	0.89	9.60	10.12	28.88

表 9：HMT 在采用 Transformer-Base 的 De→En 上的数值结果。

WMT15 De→En				***Transformer-Base***		
L	K	**CW**	**AP**	**平均滞后 DAL**		**BLEU**
-1	4	1.58	0.52	0.27	2.41	22.52
2	4	1.78	0.59	2.20	4.53	27.60
3	6	2.06	0.64	3.46	6.38	29.29
5	6	1.85	0.69	4.74	6.95	30.29
7	6	2.03	0.74	6.43	8.35	30.90
9	8	2.48	0.79	8.36	10.09	31.45
11	8	3.02	0.83	10.06	11.57	31.61
13	8	3.73	0.86	11.80	13.08	31.71

表 10：HMT 在采用 Transformer-Big 的 De→En 上的数值结果。

WMT15 德→英				***Transformer-Big***		
L	K	**CW**	**AP**	**平均滞后 DAL**		**BLEU**
-1	4	1.65	0.52	0.06	2.43	22.70
2	4	1.79	0.60	2.19	4.50	27.97
3	6	2.04	0.64	3.46	6.30	29.91
5	6	1.88	0.69	4.85	7.07	30.85
7	6	2.06	0.74	6.56	8.47	31.99
9	8	2.47	0.79	8.34	10.10	32.28
11	8	2.98	0.83	10.12	11.58	32.46
13	8	3.75	0.86	11.78	13.09	32.58