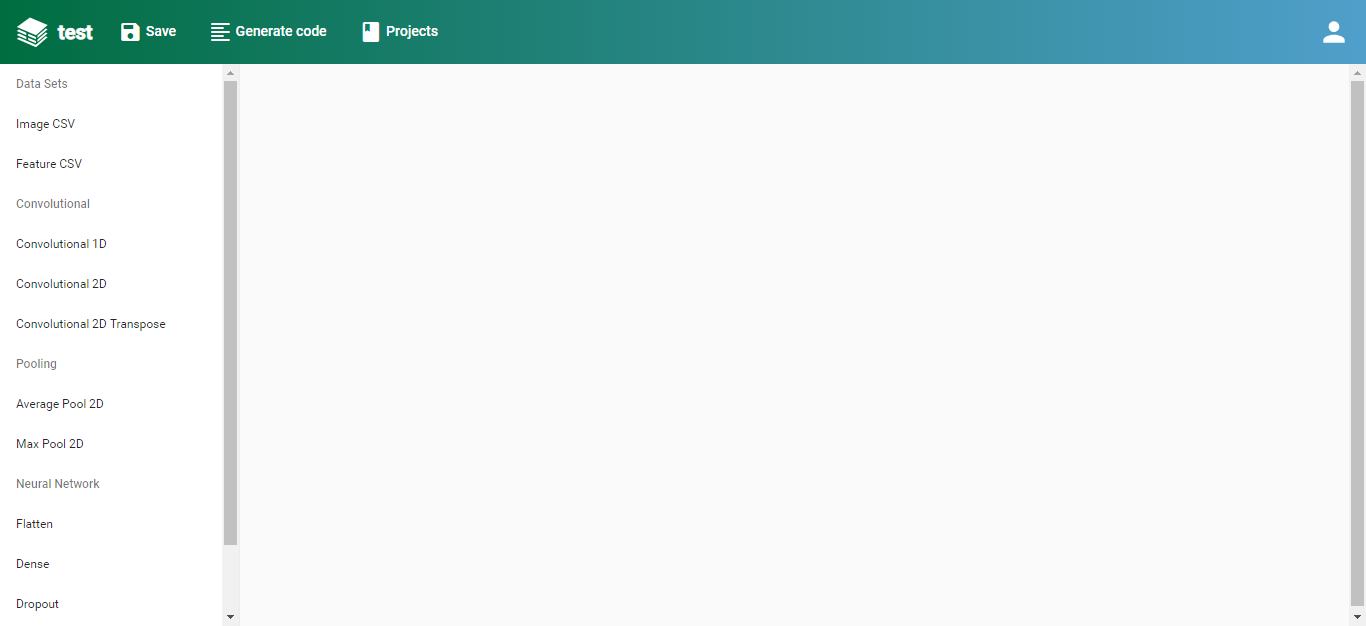
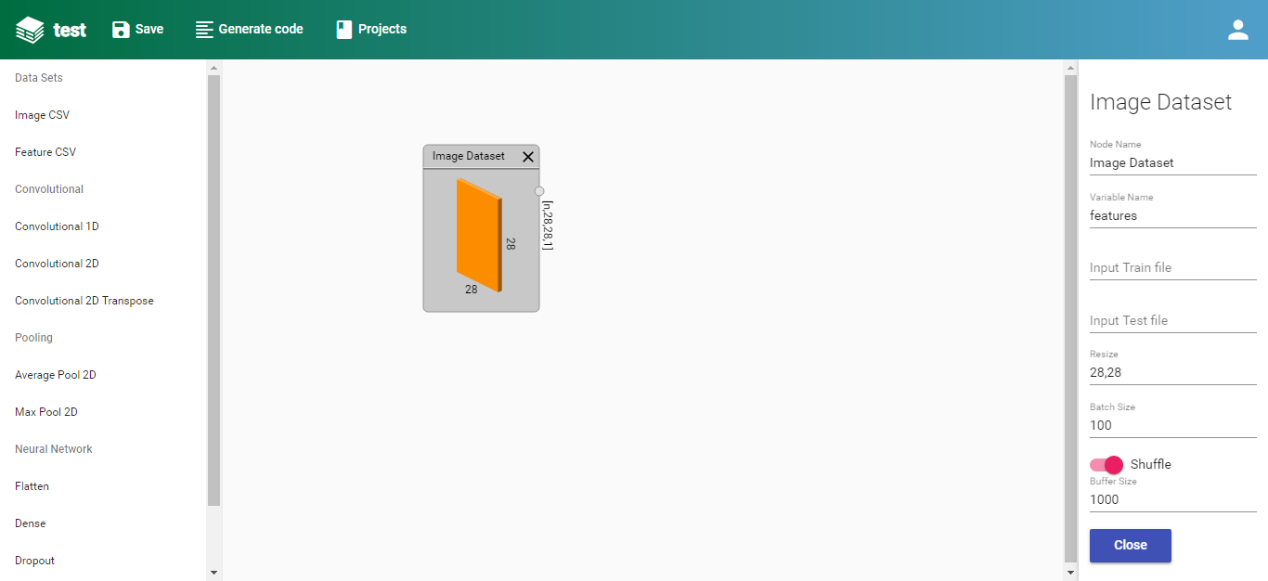
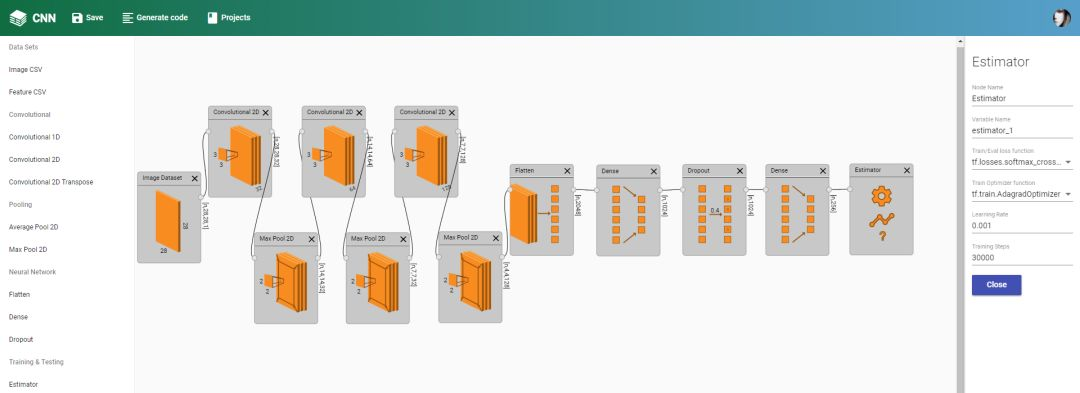
Tensoreditor：

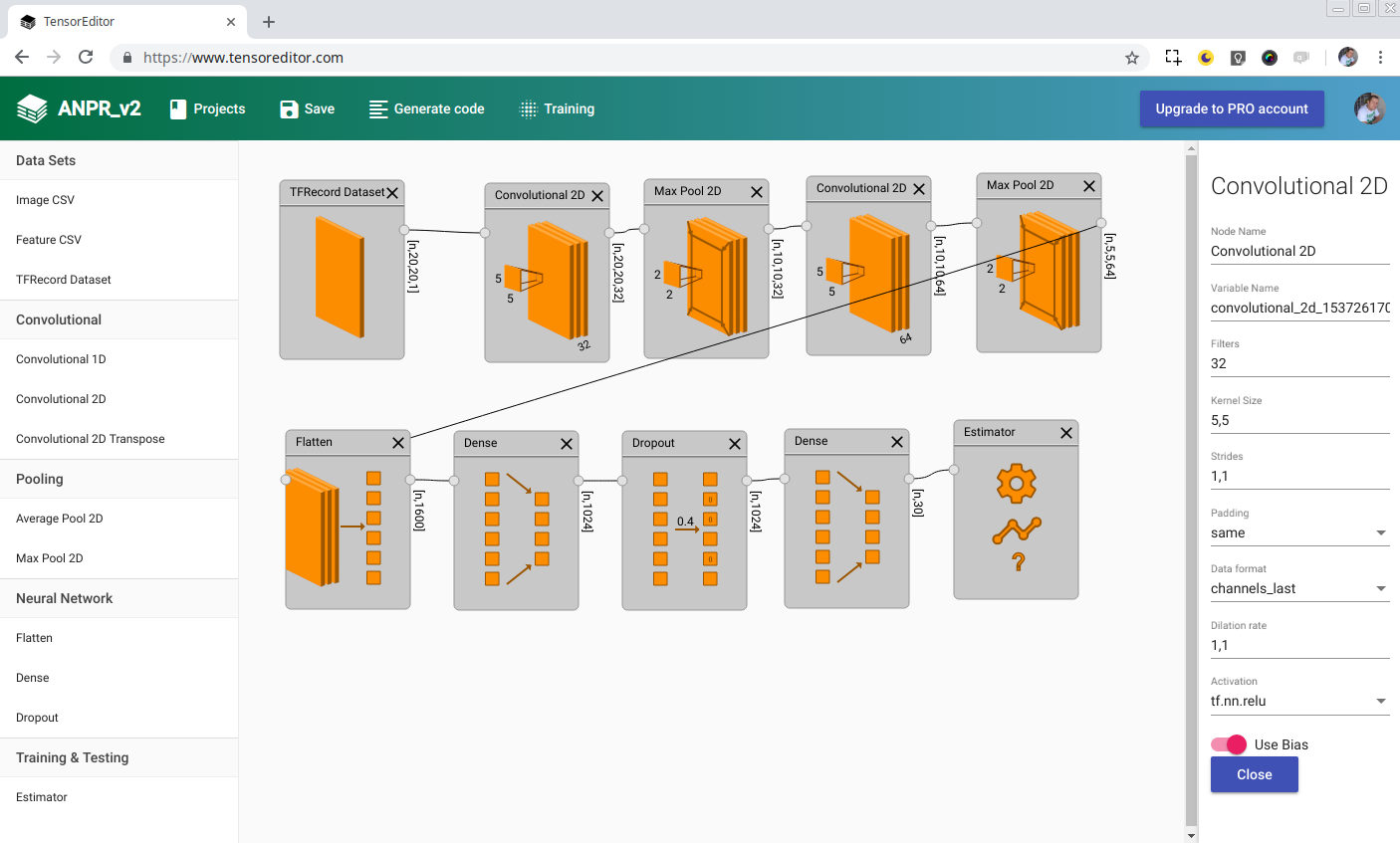
搭建神经网络的工具（2021-6域名过期）

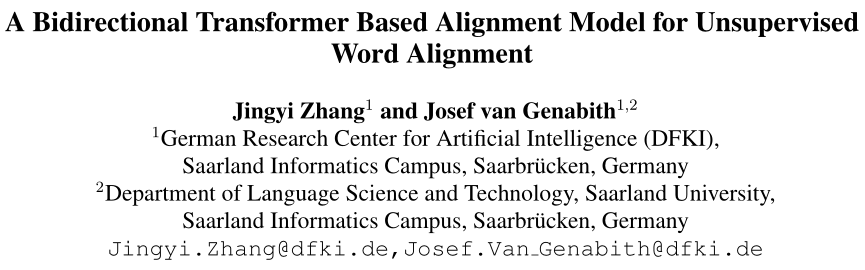
过程：











ACL2021 (CCF A)

单词对齐和机器翻译是两个密切相关的任务。神经翻译模型，如基于RNN和Transformer的模型，采用了目标-源注意机制，可以提供粗略的单词对齐，但精确度较低。高质量的词对齐可以帮助神经机器翻译在许多不同的方式，如遗漏字检测，注释转移和词汇注入。现有的单词对齐学习方法包括统计单词对齐器（如GIZA）和最近的神经单词对齐模型。

本文：提出了一种基于双向变换器的对齐（BTBA）模型，用于单词对齐任务的无监督学习。

主要贡献：

1. BTBA模型通过关注源上下文以及左侧和右侧目标上下文来预测当前目标词，从而产生准确的目标-源注意（对齐）。
2. 我们在BTBA模型中进一步微调目标到源的注意力，以使用基于全上下文的优化方法和自我监督训练获得更好的对齐。
3. 我们在三个词对齐任务上测试了我们的方法，结果表明我们的方法优于以前的神经词对齐方法和流行的统计词对齐器GIZA。

单词对齐任务定义：

有源句子和对应的目标句子。

目标是寻找两者之间的对应关系

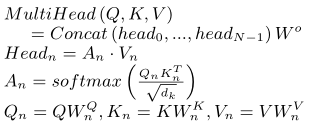
注意到可以是一对一、一对多、多对一、多对多的对齐以及未对齐的单词。

由于缺少单词对齐任务的标记训练数据（人类注释的黄金对齐），大多数单词对齐方法通过无监督学习学习单词对齐任务。

机器翻译：

通常的机器翻译步骤，是将编码器对源语句进行编码（encoder）；解码器（decoder）通过关注源上下文并执行从左到右的自回归解码来生成目标句子。在这类模型中学习到的目标-源注意可以在源词和目标词之间提供粗略的词对齐。

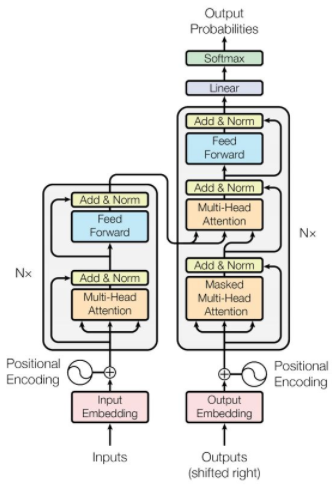
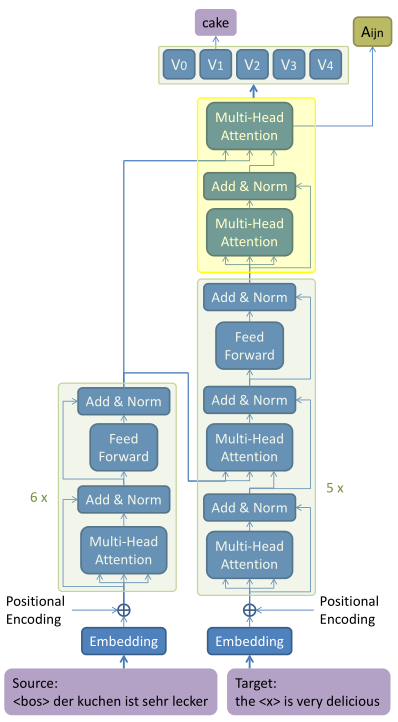
在各种翻译模型中，Transformer翻译模型在各种翻译任务中实现了最先进的结果，并且仅基于注意力机制：编码器中使用的源到源注意；解码器中使用的目标到目标和目标到源注意。变压器模型中使用的注意网络称为多头注意：



BTBA：

我们提出了一个基于双向变换的对齐（BTBA）模型，用于单词对齐任务的无监督学习。

基于BERT的启发，我们随机遮住目标句子中10%的单词，然后训练我们的BTBA模型，通过关注源上下文以及左侧和右侧目标上下文预测蒙面目标单词。因此，我们的BTBA模型可以利用左侧和右侧目标上下文来计算比原始Transformer翻译模型更准确的目标-源注意（对齐）由于从左到右的自回归解码，仅基于左侧目标上下文计算目标到源的注意。我们在BTBA模型中进一步微调目标到来源的注意力，以使用基于全上下文的优化方法和自我监督训练产生更好的对齐。



BTBA Transformer

左边用来进行encoder，和transformer完全一样。

Decoder遮住了10%，用<x>表示（这是随机的，上图只是例子。而且，因为少于10个单词，因此挡住的单词上取整，为1个）。

Decoder包括了6层：

前5层是一个self-attention（q=k=v）+一个target-to-source-attention（v=k=target，q=source）+一个feed forward。

注意点：

原始的transformer中，self-attention是只看左边的句子（图中没表示，但是decoder是需要上一个位置的向量来进行表示的，这也就意味着，encoder可以并行，但decoder只能串行）。

现在的BTBA中，self-attention是看整个句子，从而更好的进入下一步target-to-source-attn。

最后一层包含self-attention和target-to-source-attention，但没有ff子层。我们使用最后一个target-to-source-attention的输出预测被掩蔽的目标词，并使用最后一个target-to-source-attention的注意推断源词和目标词之间的词对齐。

这种想法是受到别的论文的启发，也就是强制最后的这一层去关注source单词中最重要的单词以预测target单词，因此更好的单词对齐。

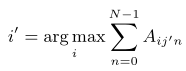
是第j个target词在最后一个target-to-source-attention中使用第n个head对第i个source的注意值。V0、V1、V2、V3、V4是decoder对5个目标字的输出（在最佳情况下，应该分别输出this,cake,is,very,delicious）。本例中，遮挡的词是cake，因此我们要学习的是，需要尝试将该值训练到最高，也就是说，输出的句子中最引起注意的（权重价值最高的）是这个词，也就是说，这个词是真正的词的可能性最大，即其包含最有用的源信息（“kuchen”）。

问题：

正因为这样，可以有效地推断目标单词“kuchen”的单词对齐方式。但是，不能为“delicious”等未屏蔽的目标词提供良好的词对齐（他们的注意力系数不够高），因为V4不用于预测任何目标词，A54n的系数也不一定大。

解决方法：

每次遮住一个词，对于长度为J的句子，遮住J次。每次收集A的值。考虑j’被遮住，计算i’，也就是对齐的位置（注意我们一开始的问题）应该是计算最大的A值（累积从第一层到第n层的注意力系数），即：



公式是从0到N-1，所以它同时考虑左侧和右侧目标上下文来计算j’的注意力。

问题：

如何利用上下文来增强target-to-source-attention呢？

解决：

在最后50步的训练中，我们不再遮住目标词，而是在显示的标出词的时候最大化最后一层的Q和K。

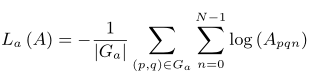
基于完整的target上下文优化Q和K可以帮助BTBA模型产生更好的注意力对齐，同时冻结其他参数可以使BTBA模型保留从蒙蔽目标词预测中学习到的知识。因为在完全基于目标上下文的优化之后，我们不需要在测试集中屏蔽目标句子。我们可以直接将原始源测试句子和目标测试句子输入BTBA模型，并计算句子中所有目标词的注意力对齐。

注：完全基于上下文的优化方法可以看作是对原始BTBA模型的fine-tune。我们根据完整的目标上下文对最后一个目标到源注意层中的两个参数Q和K进行微调，以计算更精确的单词对齐。

自监督学习：

BTBA模型通过无监督学习学习单词对齐，并且单词对齐任务不需要标记数据。我们训练两个无监督的BTBA模型，一个用于正向（源到目标）和一个用于反向（目标到源），然后使用启发式方法（如Grow-diagonal-final-and）进行对称化对齐。

因为对称化的路线比单个正向或反向模型的路线具有更好的质量。在无监督学习之后，我们使用从我们的无监督BTBA模型中获得的对称化单词对齐作为标记数据，通过以下公式来对对齐损失进行监督训练，进一步微调单词对齐任务的每个BTBA模型：



注1：在监督训练过程中，训练BTBA模型学习对齐任务，而不是蒙蔽目标词预测，因此目标句子不需要蒙蔽。

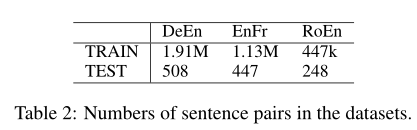
注2：单词对齐是基于单词的，不是基于字节的BPE（Byte Pair Encoding）对齐，因此在输出时需要先合并为单词，在训练时再解码成字节。（这里的字节与计算机的字节含义不同）

实验：

德语-英语 英语-法语 罗马语-英语

测试集：人工标准最佳对齐方式的单词，作为测试数据

训练集：没有单词对齐数据（即无监督），但有对应的翻译前后的句子。



模型细节：

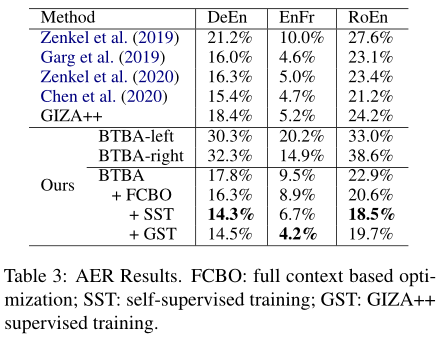
BPE可以组成40k的单词，词嵌入大小512，ff层有2048个隐单元。多头是8头。Adam。lr=0.0002.dropout=0.3。每个训练的batch包含了40k个遮挡后的句子。

单词对齐没有验证集，epoch数量依次为50,100,200。如前文所述，最后50次是明文训练。

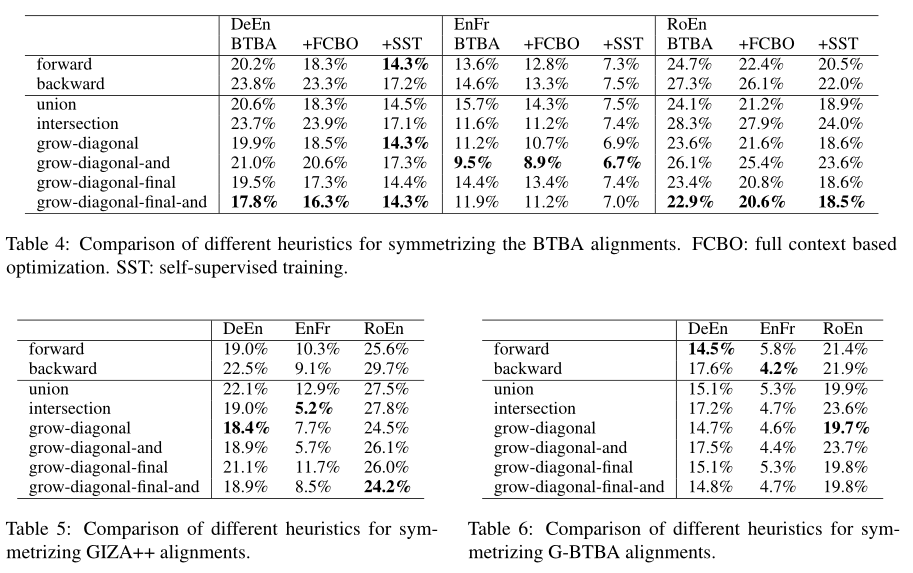
比较了以下对齐方式，即启发式算法：grow-diagonal,grow-diagonal-final,grow-diagonal-final-and,grow-diagonal-and.

Baseline: GIZA++

指标：AER-Alignment Error Rate



启发式算法选择：



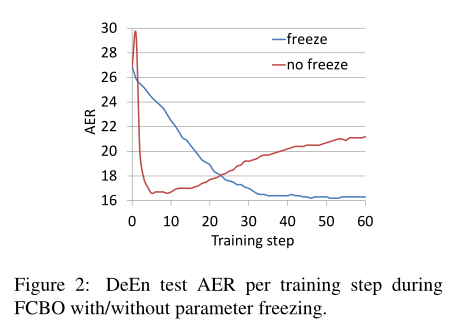
其中，

BTBA：grow-diagonal-final-and在DeEn and RoEn最好。grow-diagonal-and在EnFr 最好.

GIZA++和G-BTBA：grow-diagonal-final-and在DeEn and RoEn 最好。grow-diagonal-and在EnFr 最好。但是，不同算法的选择差距比BTBA大，也就是说，BTBA更不受这些启发式的影响。

FCBO（全上下文参与的优化）中参数冻结的意义：

可以冻结前n层的参数，也可以不冻结。德语-英语的转换结果如下：



可以看到，不冻结的AER结果（红色曲线）先略有增加，然后急剧下降，很快又增加。相比之下，当我们冻结大多数参数时，AER结果（蓝色曲线）稳定下降，最终得到的结果（16.3%）比不冻结参数（16.7%）更好。

注：图2中的结果是基于完整的目标上下文计算的，即目标句没有被掩盖。由此验证以下结论：

1. 不带FCBO的BTBA模型只能用于推断屏蔽目标词的词对齐。在没有FCBO的情况下，使用BTBA模型推断未屏蔽目标词（即剩余词）的单词对齐与使用BTBA模型推断屏蔽目标词的单词对齐（如表3所示，为17.8%）相比，AER结果较差（如图2所示，为26.9%）。
2. FCBO可以快速改进使用BTBA模型推断未屏蔽目标单词对齐的结果，最终在FCBO之后，BTBA模型可以有效地使用完整的目标上下文来计算比没有FCBO的原始BTBA模型更好的单词对齐（16.3%对17.8%，如表3所示）。

监督数据：

从我们的无监督BTBA模型推断出的单词对齐作为标记数据，通过监督训练进一步微调对齐任务的每个单向BTBA模型。

我们还测试了使用单向BTBA比对而不是对称BTBA比对作为监督训练的标记数据。

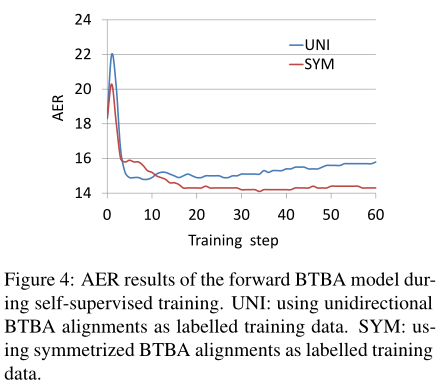
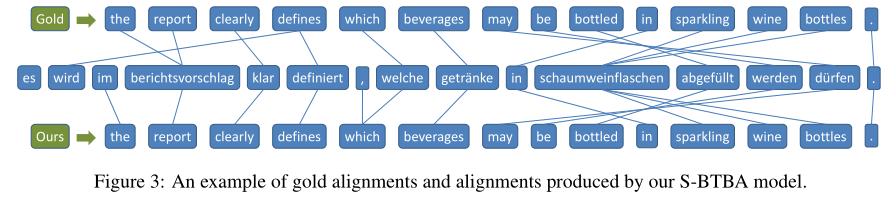


图4（蓝色曲线）显示了DeEn任务的前向BTBA模型的输出作为指标（不考虑对齐对称的改进的话），其得到的准确率还是很高。当然，如果使用了对称对齐化的话，效果会更好（红色曲线）。

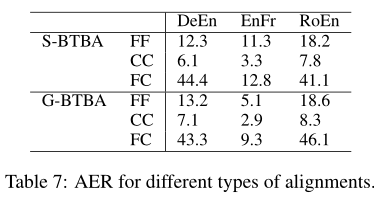
此外，监督训练可以改进BTBA模型，即使标记训练数据的质量比BTBA模型本身稍差，例如，对于Ro-En任务，通过监督训练，使用GIZA++对前向BTBA模型进行微调，改善了前向BTBA模型的结果（22.4%→21.4%）。尽管，使用GIZA++模型进行单词对齐的效果是比前向BTBA还要差。（相当于1+1>2的效果）

对齐错误分析（个案分析）：

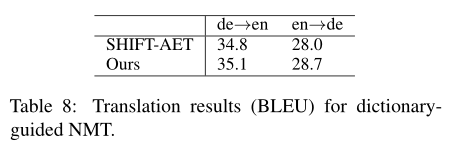
我们分析了系统产生的对齐错误，发现大多数对齐错误是由虚词引起的。如图中的对齐示例所示，源和目标对应的内容词（例如“definiert”和“defines”）都通过我们的模型正确对齐，但是诸如“the”、“im”和“wird”之类的虚词没有正确对齐。



为了进行更详细的分析，我们计算了我们模型的3种不同对齐类型的AER结果：FF（两个虚词之间的对齐）、CC（两个内容词之间的对齐）和FC（虚词和内容词之间的对齐）。结果显示，我们的模型在CC路线上取得的结果明显优于FF和FC路线。虚词比内容词更难对齐，因为平行句对中的内容词通常有非常清晰的对应关系（如图3中的“defines”与“definiert”明确对应），但虚词（如“the”、“es”和“im”）不一致在并列句中使用较为灵活，且没有明确的对应词，这显著增加了对齐难度。



对其它神经网络的改进（即看作Pre-trained model）：



对于下游任务，单词对齐可用于改进字典指导的NMT。在NMT中的每个解码步骤中，需要使用SHIFT-AET方法计算新生成的目标词的对齐度，然后通过鼓励词典中预先指定的翻译来修改新生成的目标词。SHIFT-AET校准方法的目标是将一个单独的校准模块添加到原来的Transformer翻译模型中，并使用原来的Transformer中的注意力权重产生的校准来训练单独的校准模块。

现在，将BTBA模型推断的对齐作为标记数据，用于监督SHIFT-AET对齐模块，并对德-英进行字典指导翻译。表格中给出了字典指导的NMT的翻译结果，并表明我们的对齐方法比原来的SHIFT-AET方法具有更高的翻译质量。

BPE流程：

1. 确定subword词表大小
2. 统计每一个连续字节对的出现频率，并保存为code\_file。这个是git中learn-bpe完成

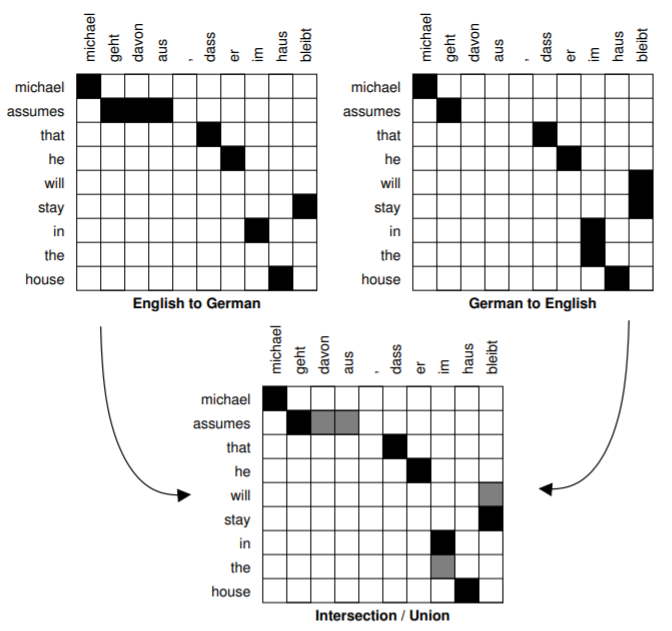
3． 将单词拆分为字符序列并在末尾添加后缀“ </w>”，而后按照code\_file合并新的subword，首先合并频率出现最高的字节对。例如单词birthday，分割为['b', 'i', 'r', 't', 'h', 'd', 'a', 'y</w>']，查code\_file，发现'th'出现的最多，那么合并为['b', 'i', 'r', 'th', 'd', 'a', 'y</w>']，最后，字符序列合并为['birth', 'day</w>']。然后去除'</w>',变为['birth', 'day']，将这两个词添加到词表。这个是apply-bpe完成。

4. 重复第3步直到达到第2步设定的subword词表大小或下一个最高频的字节对出现频率为1

Grow-diagonal-final-and：

问题：如何标记对齐？

传统方法：求union或intersection



问题：还有更多的对齐没有表示出来吗？

新的方法：利用邻居的节点扩张来完成对齐操作。

所谓邻居（neighbor），是指距离为1的点，共8个。

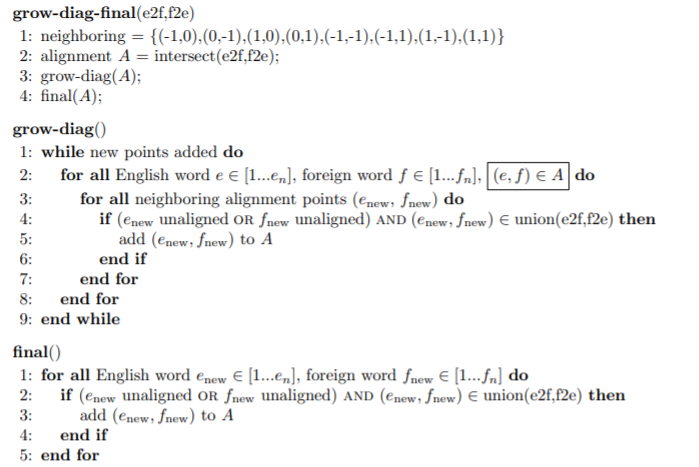
现在，从交集intersection开始进行扩展：

遍历每一个点，如果其邻居是在union中并且这一行或这一列还没有对齐的位置，就将其标为对齐的位置。

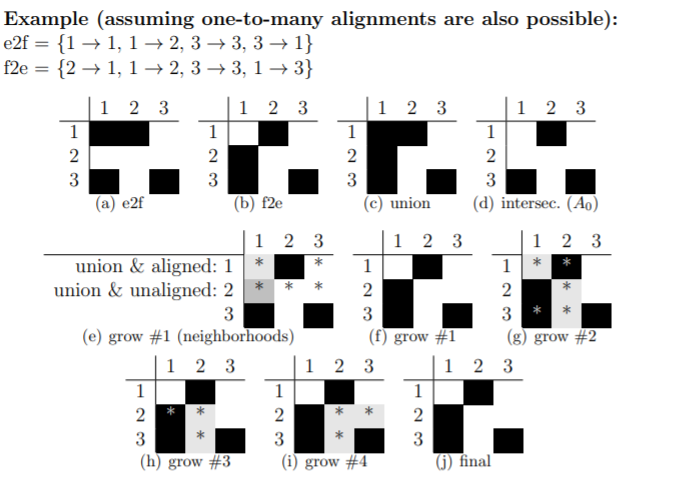
循环到没有新的点加入为止。

最后，观察每个点（共n\*n个），如果所在行**或**列没有对齐且在union中，那么加上该点作为对齐点。

代码：



例子：



放缩：将或改为和，即得到Grow-diagonal-final-and方法。