Testing Machine Translation via Referential Transparency

191250205 赵喆德

目 录

CONTENT

Part 1

INTRODUCTION

机器翻译软件的自动化测试是一个有待解决的问题，而且非常具有挑战性。首先，与传统的软件相比，神经机翻译软件的逻辑很大程度上嵌入在后端模型的结构和参数中，因此现有的基于代码的测试技术不能直接用于测试 NMT；其次，现有的AI软件测试方法主要针对更简单的用例（如10-class分类），相比之下，测试机器翻译是一项更困难的任务：一个源文本可以有多个正确的翻译，并且输出空间要大得多；第三，现有的机器翻译测试技术通过语言模型替换一个句子中的一个单词来生成测试用例（即合成句子），因此它们的表现受到现有语言模型的熟练程度的限制。

Part 1

INTRODUCTION

为了解决这个具有挑战性的问题，作者引入 RTIs（引用透明性输入），这是一个用于评价机器翻译软件的新颖而通用的概念。RTI 的核心思想来自于编程语言（尤其是函数式编程）中的一个概念，即引用透明性：方法应该总是为给定的参数返回相同的值。在本文中，将 RTI 定义为在不同上下文中具有固定翻译的文本。关键是生成包含相同 RTI 的一对文本，并检查这对文本的翻译是否不变。为了实现这个概念，作者实现了 Purity，一个从任意未标记的句子中提取短语的工具，具体是，给定一个源句，Purity 通过 constituency parser 提取短语，并通过将 RTI 与其包含的句子或短语分组来构造 RTI 对。如果同一 RTI 的译文之间差异很大，则将这对文本及其译文报告为一个可疑的问题。本文的核心思想在概念上不同于现有的方法，现有方法的上下文是固定的，只替换了某个词，并假设翻译应该只有小的变化；相反，本文假设 RTI 的翻译在不同的句子/短语之间应该是不变的，上下文可变。

Source text	Target text	Target text meaning	
In a series of pointed tweets, Trump said the economy is good, so Chevy should keep the plant, the closure of which was announced back in November.	特朗普在一系列明确的推文中表示，经济状况良好，因此雪佛兰应保留该工厂，该工厂已于 11 月宣布关闭。(by Google)	In a series of clear tweets, Trump said the economy is good, so Chevy should keep the plant, the closure of which was announced back in November.	✗
a series of pointed tweets	一系列针对性的推文 (by Google)	a series of pointed tweets	✓
Holmes in a movie based on Bad Blood	福尔摩斯在电影的基础上坏血 (by Bing)	Holmes' blood becomes bad based on a movie	✗
a movie based on Bad Blood	一部基于坏血的电影 (by Bing)	a movie based on Bad Blood	✓

Part 02

BACKGROUND

引用透明性：输出的结果只由输入决定。在编程语言领域，引用透明性是指在不改变程序结果的情况下，一个表达式可以被其在程序中的相应值所替换的能力。例如，数学函数（如平方根函数）是引用透明的，而打印时间戳的函数则不是。平方根只与输入有关。

RTI：定义为在不同上下文中具有固定翻译的文本。

RTI 定义为跨文本（句子和短语）翻译不变的一段文本。

关键是生成包含相同的RTI的一堆文本，并检查这一对文本的翻译是否不变。作者实现了Purity，一个从人为标记的句子中提取短语的工具。

Purity实现原理：给定一个源句,Purity通过constituency parser提取短语，并通过将RTI与其包含的句子或短语分组来构造RTI对。

Part 03

Purity实现原理

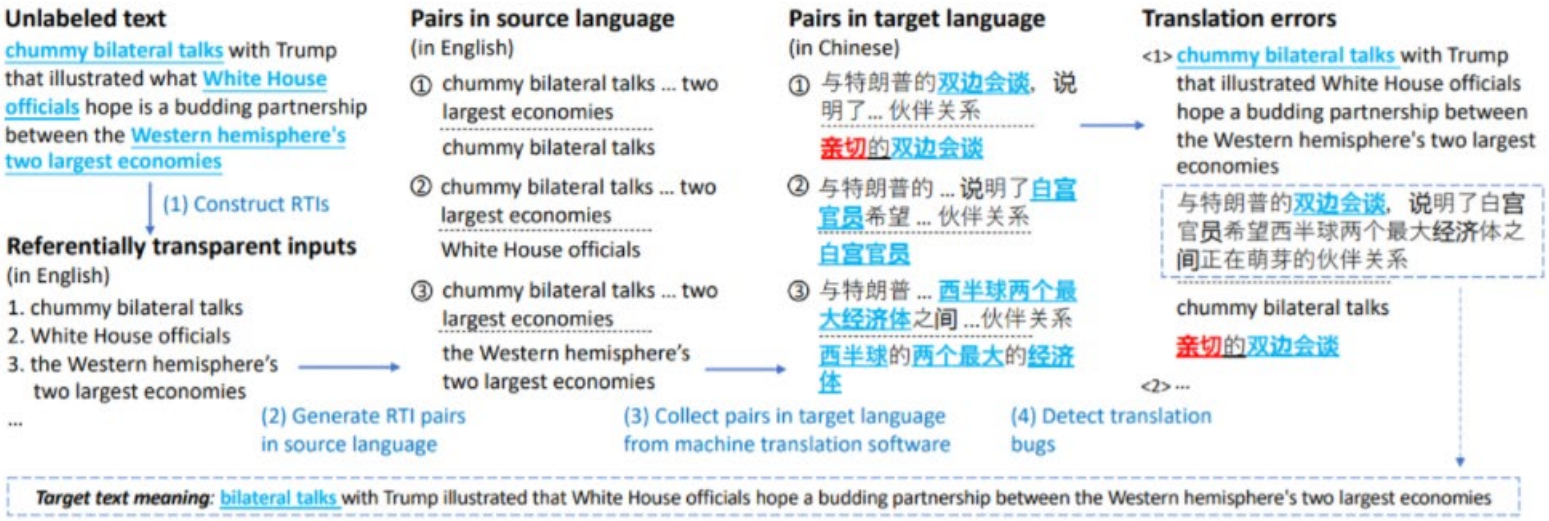


Figure 2: Overview of our RTI implementation. We use one English phrase as input for clarity and simplicity.

(1) Identifying referentially transparent inputs 分析每个句子的成分，提取短语列表作为 RTIs；

(2) Generating pairs in source language 将每个短语与包含它的短语或原句配对，形成 RTI 对。

(3) Collecting pairs in target language 将 RTI 对输入被测机器翻译软件，收集相应的译文。

(4) Detecting translation errors 比较每一对的 RTI 对的译文，如果 RTI 的翻译之间存在很大的差异，则认为这对翻译可能存在翻译错误。

Part 03

PURITY'S IMPLEMENTATION

RTI实现的伪代码

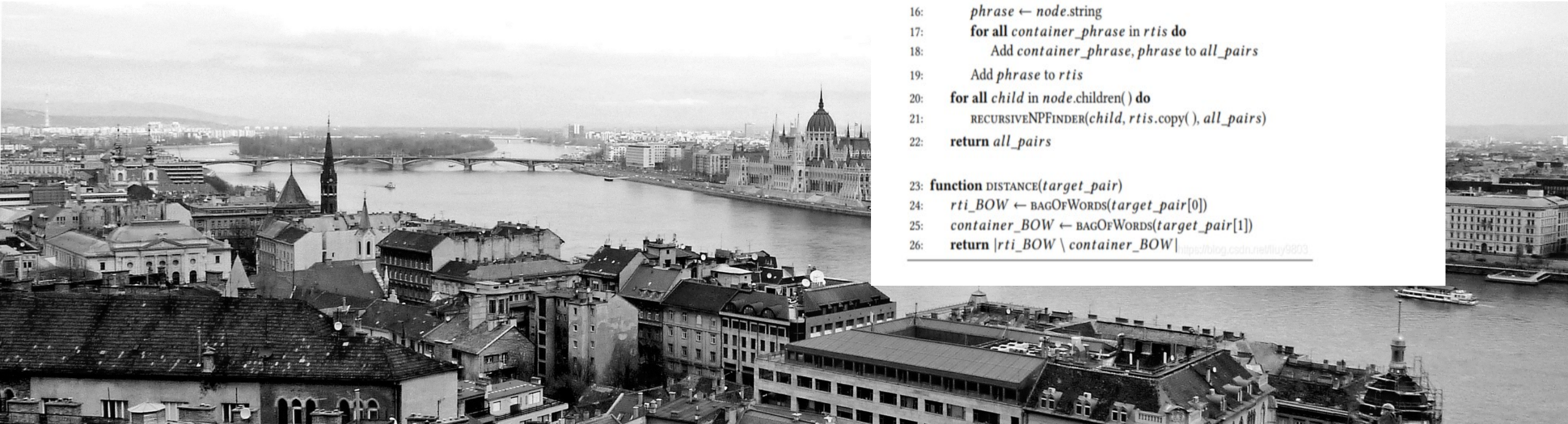
Algorithm 1 RTI implemented as *Purity*.

Require: *source_sents*: a list of sentences in source language
d: the distance threshold
Ensure: *suspicious_issues*: a list of suspicious pairs

```
1:  suspicious_issues ← List( )                    ▷ Initialize with empty list
2:  for all source_sent in source_sents do
3:      constituency_tree ← PARSE(source_sent)
4:      head ← constituency_tree.head( )
5:      RTI_source_pairs ← List( )
6:      RECURSIVENPFINDER(head, List( ), RTI_source_pairs)
7:      RTI_target_pairs ← TRANSLATE(RTI_source_pairs)
8:      for all target_pair in RTI_target_pairs do
9:          if DISTANCE(target_pair) > d then
10:             Add source_pair, target_pair to suspicious_issues
11: return suspicious_issues

12: function RECURSIVENPFINDER(node, rtis, all_pairs)
13:     if node is leaf then
14:         return
15:     if node.constituent is NP then
16:         phrase ← node.string
17:         for all container_phrase in rtis do
18:             Add container_phrase, phrase to all_pairs
19:         Add phrase to rtis
20:     for all child in node.children( ) do
21:         RECURSIVENPFINDER(child, rtis.copy( ), all_pairs)
22:     return all_pairs

23: function DISTANCE(target_pair)
24:     rti_BOW ← BAGOFWORDS(target_pair[0])
25:     container_BOW ← BAGOFWORDS(target_pair[1])
26:     return |rti_BOW \ container_BOW|
```

Part 03

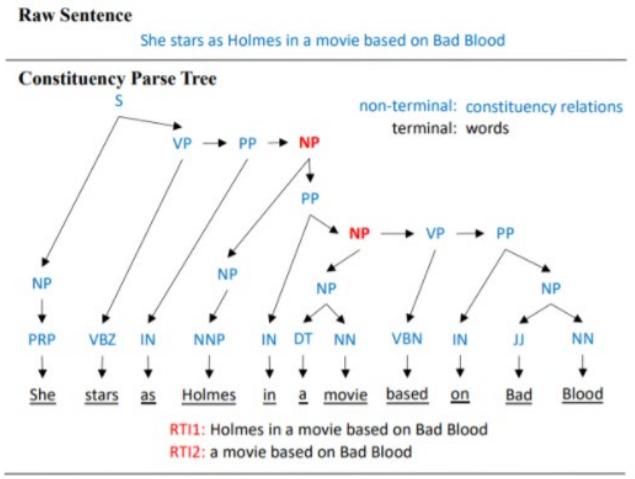


Figure 3: A constituency parse tree example. The non-terminal nodes in bold and red are the RTIs extracted by our approach.

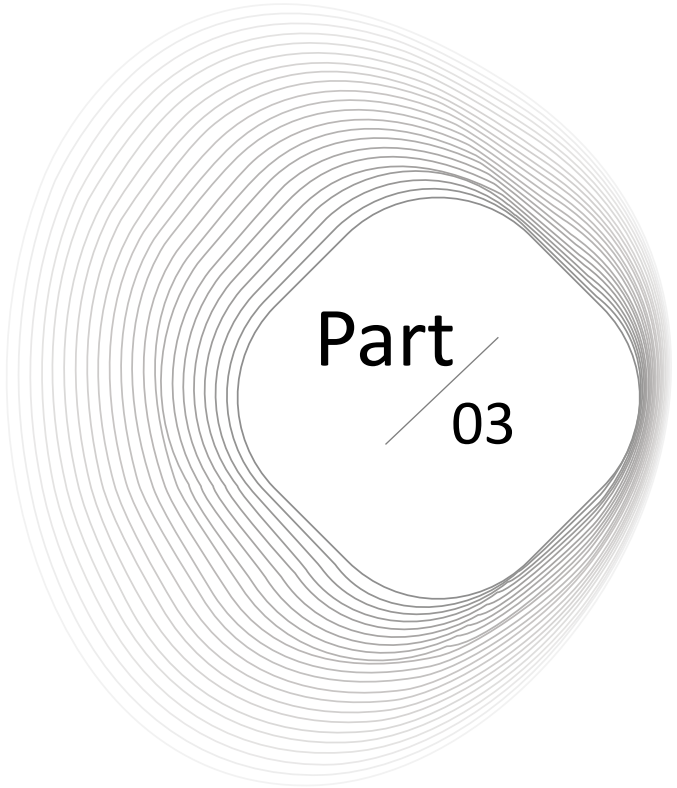
工具实现总流程

3.1 Identifying RTIs

3.1.1 收集RTIs列表，即找到具有独特意义的文本片段，这个意义应该在不同的上下文都适用（相同的翻译场景）。本文只考虑名词短语（从源语言的一组句子中提取名词短语）。

3.1.2使用NLP工具 constituency parser 输出字符串的句法树结构，非叶节点是成分结构关系，叶节点是单词。成分结构关系（将一个句子解析为一系列成分结构，即对句子进行层级结构分析。如主语+谓语，名词短语+动词短语，不断解析直到不能分出更小的成分为止）

3.1.3通常一个RTI可以包含另一个较短的RTI。从一个句子中获得所有的名词短语后，根据经验过滤掉包含十个以上和三个以下的非停止词的词。(这里只考虑了中文中常见的停止词)。



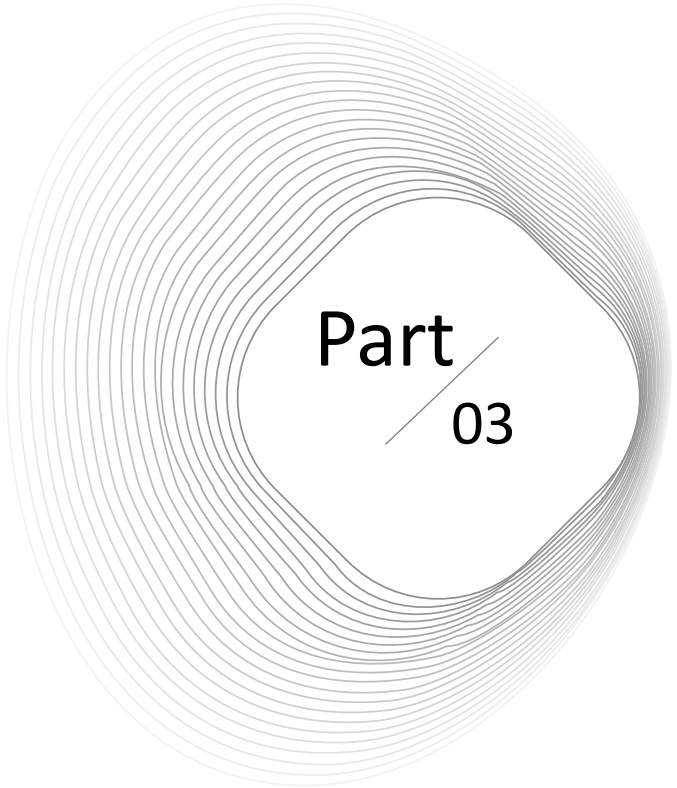
工具实现总流程

3.2 Generating Pairs in Source Language

生成的 RTIs 列表必须与包含短语配对，这些短语将用于引用透明性验证。具体来说，每个 RTI 对应该有两个包含相同短语的不同文本片段。将 RTI 与发现它的全文和同一个句子所有包含它的 RTI 配对，构建3对 RTI：(1) RTI1 和原句、(2) RTI2 和原句、(3) RTI1 和 RTI2。

3.3 Collecting Pairs in Target Language

一旦得到一组RTI对，下一步是将这些文本（以给定的源语言）输入到被测机器翻译软件中，并收集它们的翻译（以任何选定的目标语言）。使用谷歌和 Bing 提供的 api，这些 api 返回的结果与二者的 Web 接口返回的内容相同。在复现时额外进行了对其他翻译库的调用。



工具实现总流程

3.4 Detecting Translation Errors

3.4.1NLP的单词对齐技术，可以将源文本中的单词/短语映射到目标文本中的单词/短语，然而现有工具的性能很差。所以采用词袋BoW模型。

3.4.2 BoW词袋模型：最早出现在自然语言处理和信息检索邻域。该模型忽略掉文本的语法和语序等要素，将其仅仅看作是若干个词汇的集合，文档中每个单词的出现都是独立的。BoW使用一组无序的单词来表达一段文字或一个文档。exp：一句话由n个单词组成，分别计算每个单词出现的频率，将其构建成为向量，向量维数由单词个数构成。

3.4.3 使用了特殊规定的方法计算两个词袋之间的距离。

Part 4

EVALUATION

Experimental Setup and Dataset

将 Purity 与两种 sota 方法 SIT、TransRepair (ED) 进行比较。Purity 使用 CNN 文章中收集的数据集，详细信息如表1所示，这个数据集包含两个语料库：Politics 和 Business，使用两类语料是因为想要评估不同语境下句子的 Purity 性能。

Precision on Finding Erroneous Issues

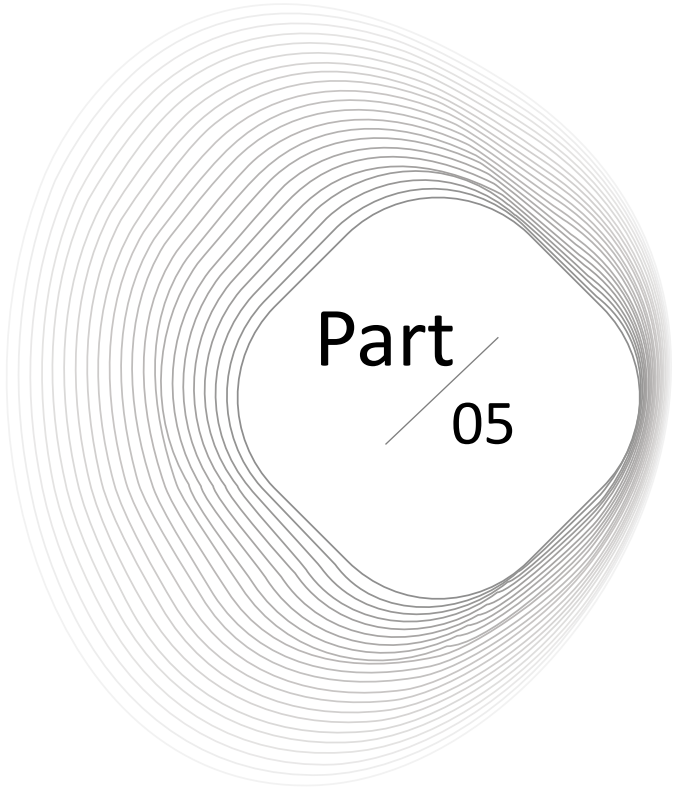
准确率评估，即有多少报告的问题包含实际的翻译错误。

Unique Erroneous Translation

可以看到，当 d = 0 时，Purity 发现了54个~74个翻译错误，设置更大的距离阈值以获得更高的准确率，得到更少的错误翻译。

Translation Error Reported

Purity 能够发现各种各样的翻译错误，在实验中成功地发现了5种翻译错误：欠翻译、过度翻译、单词/短语误译、修饰错误和逻辑不清。



Conclusion

作者提出了一个简单、通用的概念——引用透明输入 RTI——用于测试机器翻译软件，不同于现有的——扰乱一个自然句中的词（上下文固定），并假设翻译应该只有小的变化——的方法，本文假设 RTIs 应该在不同的上下文中具有不变的翻译，因此 RTI 可以发现不同种类的翻译错误，从而补充现有的方法。测试谷歌翻译和微软必应翻译，分别发现了123和142个错误的翻译，且目前用时最短，证明了 Purity 测试机器翻译软件的能力。在未来的工作中，作者将继续细化一般方法，并将其扩展到其他 RTI 实现方式，例如将动词短语作为 RTIs 或将整个句子作为 RTIs，将它们与语义上不相关的句子相连配对，还将在翻译错误诊断和机器翻译系统的自动修复方面开展广泛的工作。

THANKS

191250205 赵喆德