1. BLEU

https://blog.csdn.net/qq_30232405/article/details/104219396

BLEU的计算步骤：

1. **n-gram匹配**：
   - BLEU的核心思想是比较翻译结果和参考翻译之间的n-gram（n个连续单词的组合）的重合情况。
   - 一般会计算1-gram（单个词）、2-gram、3-gram和4-gram的重合率。
   - 对于每个n-gram，计算机器翻译中出现的n-gram和参考翻译中出现的n-gram的交集。
2. **精确度（Precision）**：
   - 对于每个n-gram，计算机器翻译中的n-gram出现次数与参考翻译中n-gram的出现次数之间的比率。
   - 例如，假设参考翻译中某个n-gram出现了3次，而机器翻译中该n-gram出现了2次，那么该n-gram的精确度为2/3。
3. **惩罚项（Brevity Penalty，BP）**：
   - BLEU引入惩罚项来解决翻译长度与参考翻译长度不一致的问题。机器翻译输出如果比参考翻译短，会受到惩罚。
   - **BP的计算方式**：

$$BP = \begin{cases} 1 & \text{if translated length} > \text{reference length} \\ \exp(1 - \frac{\text{reference length}}{\text{translated length}}) & \text{if translated length} \leq \text{reference length} \end{cases}$$

4. **最终得分**：
   - 最终的BLEU得分是所有n-gram精确度的加权平均值，再乘以惩罚项：

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

其中，p_n 是n-gram的精确度，w_n 是权重（通常为1/N），N 表示计算的n-gram最大值（一般为4）。

1. 计算n-gram匹配：

我们以1-gram、2-gram和3-gram为例来计算匹配情况。

1-gram匹配：

- 参考翻译中的1-gram：["The", "cat", "sits", "on", "the", "mat"]
- 机器翻译中的1-gram：["The", "cat", "is", "on", "the", "mat"]

计算机器翻译中与参考翻译1重合的词：

- 重合词有：["The", "cat", "on", "the", "mat"]
  重合数 = 5。

所以1-gram的精确度 $p_1 = \frac{5}{6} = 0.8333$。

2-gram匹配：

- 参考翻译中的2-gram：["The cat", "cat sits", "sits on", "on the", "the mat"]
- 机器翻译中的2-gram：["The cat", "cat is", "is on", "on the", "the mat"]

计算机器翻译中与参考翻译1重合的2-gram：

- 重合的2-gram有：["The cat", "on the", "the mat"]
  重合数 = 3。

所以2-gram的精确度 $p_2 = \frac{3}{5} = 0.6$。

3-gram匹配：

- 参考翻译中的3-gram：["The cat sits", "cat sits on", "sits on the", "on the mat"]
- 机器翻译中的3-gram：["The cat is", "cat is on", "is on the", "on the mat"]

计算机器翻译中与参考翻译1重合的3-gram：

- 重合的3-gram有：["on the mat"]
  重合数 = 1。

所以3-gram的精确度 $p_3 = \frac{1}{4} = 0.25$。

2. 计算惩罚项（Brevity Penalty，BP）：

假设机器翻译的长度为6个词，而参考翻译的长度为7个词。

计算BP：

$$BP = \exp(1 - \frac{7}{6}) = \exp(1 - 1.1667) = \exp(-0.1667) \approx 0.847$$

3. 计算最终BLEU得分：

假设我们使用1-gram、2-gram和3-gram的加权平均，权重为 $\frac{1}{3}$。

计算BLEU得分：

$$BLEU = BP \times \exp\left(\frac{1}{3}\log(0.8333) + \frac{1}{3}\log(0.6) + \frac{1}{3}\log(0.25)\right)$$

$$BLEU = 0.847 \times \exp\left(\frac{1}{3} \times (-0.1823 + -0.5108 + -1.3863)\right)$$

$$BLEU = 0.847 \times \exp(-0.6931)$$

$$BLEU = 0.847 \times 0.5 \approx 0.4235$$

优势：自动+快

劣势：语法和语义关注不够，对于长度比较敏感

2. ROUGE

https://blog.csdn.net/m0_37531129/article/details/102809855

ROUGE-L的计算方法

ROUGE-L的计算包括以下几个步骤：

1. **计算最长公共子序列（LCS）**：找出生成文本和参考文本之间的最长公共子序列。
2. **计算召回率（Recall）、精确度（Precision）和F1分数**：
   - 召回率（Recall）：生成文本的LCS与参考文本的LCS长度之比。
   - 精确度（Precision）：生成文本的LCS与生成文本的长度之比。
   - F1分数：精确度和召回率的调和平均。

ROUGE-L公式：

- **Recall**：

$$\text{Recall} = \frac{\text{LCS Length}}{\text{Reference Length}}$$

- **Precision**：

$$\text{Precision} = \frac{\text{LCS Length}}{\text{Generated Length}}$$

- **F1-Score**：

$$\text{F1} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$

$\text{Recall} + \text{Precision}$

例子

假设我们有以下的参考文本和生成文本：

- **参考文本**:
  "The cat sat on the mat."
- **生成文本**:
  "The cat is on the mat."

1. 计算LCS（Longest Common Subsequence）

首先，我们需要找到参考文本和生成文本之间的最长公共子序列。可以通过比对两个句子的单词来确定LCS。

- **参考文本**: "The cat sat on the mat."
- **生成文本**: "The cat is on the mat."

最长公共子序列为：

- **LCS** = "The cat on the mat"
  LCS的长度为 **5**（共有5个单词）。

2. 计算召回率（Recall）

召回率是LCS的长度除以参考文本的长度。参考文本中有6个单词。

- **召回率**:

$$\text{Recall} = \frac{\text{LCS Length}}{\text{Reference Length}} = \frac{5}{6} \approx 0.8333$$

3. 计算精确度（Precision）

精确度是LCS的长度除以生成文本的长度。生成文本中有6个单词。

- **精确度**:

$$\text{Precision} = \frac{\text{LCS Length}}{\text{Generated Length}} = \frac{5}{6} \approx 0.8333$$

4. 计算F1分数

F1分数是召回率和精确度的调和平均值。

- **F1分数**:

$$\text{F1} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} = \frac{2 \times 0.8333 \times 0.8333}{0.8333 + 0.8333} \approx 0.8333$$

优势：自动+快

劣势：语法和语义关注不够

3. KL散度

```
p_dist = {
   "the": 2/5,
   "cat": 1/5,
   "sat": 1/5,
   "on": 1/5,
   "mat": 1/5
}

q_dist = {
   "the": 2/5,
   "cat": 1/5,
   "sat": 1/5,
```

"on": 1/5,
"rug": 1/5
}

对单词 "the"：

- $P(\text{the}) = \frac{2}{5}, Q(\text{the}) = \frac{2}{5}$
- $P(\text{the}) \log \frac{P(\text{the})}{Q(\text{the})} = \frac{2}{5} \log \frac{\frac{2}{5}}{\frac{2}{5}} = \frac{2}{5} \log 1 = 0$

对单词 "cat"：

- $P(\text{cat}) = \frac{1}{5}, Q(\text{cat}) = \frac{1}{5}$
- $P(\text{cat}) \log \frac{P(\text{cat})}{Q(\text{cat})} = \frac{1}{5} \log \frac{\frac{1}{5}}{\frac{1}{5}} = \frac{1}{5} \log 1 = 0$

对单词 "sat"：

- $P(\text{sat}) = \frac{1}{5}, Q(\text{sat}) = \frac{1}{5}$
- $P(\text{sat}) \log \frac{P(\text{sat})}{Q(\text{sat})} = \frac{1}{5} \log \frac{\frac{1}{5}}{\frac{1}{5}} = \frac{1}{5} \log 1 = 0$

对单词 "on"：

- $P(\text{on}) = \frac{1}{5}, Q(\text{on}) = \frac{1}{5}$
- $P(\text{on}) \log \frac{P(\text{on})}{Q(\text{on})} = \frac{1}{5} \log \frac{\frac{1}{5}}{\frac{1}{5}} = \frac{1}{5} \log 1 = 0$

对单词 "mat"：

- $P(\text{mat}) = \frac{1}{5}, Q(\text{mat}) = 0$ (生成文本中没有这个单词，概率为0)
- 由于生成文本中没有这个单词，在计算时要避免除零错误，假设 $Q(\text{mat}) = 1e-10$:
- $P(\text{mat}) \log \frac{P(\text{mat})}{Q(\text{mat})} = \frac{1}{5} \log \frac{\frac{1}{5}}{1e-10} \approx \frac{1}{5} \log 2e+9 \approx 0.0201$

对单词 "rug"：

- $P(\text{rug}) = 0, Q(\text{rug}) = \frac{1}{5}$ (参考文本中没有这个单词，概率为0)
- 同理，假设 $P(\text{rug}) = 1e-10$:
- $P(\text{rug}) \log \frac{P(\text{rug})}{Q(\text{rug})} = 1e-10 \log \frac{1e-10}{\frac{1}{5}} \approx 0.0000$

步骤 3：求和并得到KL散度

最后，将上述结果相加：

$$D_{KL}(P \parallel Q) \approx 0 + 0 + 0 + 0 + 0.0201 + 0.0000 = 0.0201$$

$$D_{KL}(P \parallel Q) \approx 0 + 0 + 0 + 0 + 0.0201 + 0.0000 = 0.0201$$

4. METEOR

想比BLEU加了同义词匹配，词形还原，词序因素

https://blog.csdn.net/pearl8899/article/details/112452652

METEOR的计算流程

METEOR的核心思想是对翻译（或生成文本）和参考文本之间的匹配进行多维度的评估，而不是单纯依赖于单词的精确匹配。它使用精确度、召回率、同义词匹配等来综合衡量文本的质量。

METEOR的计算分为以下几个步骤：

1. 同义词匹配（Synonym Matching）

METEOR通过查找生成文本和参考文本中的同义词来扩展匹配的单词。在英语中，使用WordNet等词典来查找同义词，如果生成文本中的单词与参考文本中的同义词匹配，则认为它们是匹配的。

2. 精确匹配（Exact Matching）

精确匹配是最基本的匹配方式，指生成文本中的单词与参考文本中的单词完全一致。精确匹配的单词对质量评估起到重要作用。

3. 词形还原匹配（Stemming Matching）

词形还原匹配指的是通过词形还原（如将动词的不同形式转化为原形）来进行匹配。例如，“running”和“run”会被视为匹配的单词。

4. 词序（Word Order）

为了考虑词序对文本质量的影响，METEOR会惩罚生成文本中单词顺序与参考文本中的顺序不一致的情况。这个惩罚是通过计算对齐的单词对位置差异来实现的。

5. 计算精确度和召回率

- **精确度 (Precision)**：在生成文本中匹配的单词数占生成文本总单词数的比例。
- **召回率 (Recall)**：在参考文本中匹配的单词数占参考文本总单词数的比例。

精确度和召回率的计算公式为：

$$\text{Precision} = \frac{\text{Matched Words}}{\text{Generated Words}}$$

$$\text{Recall} = \frac{\text{Matched Words}}{\text{Reference Words}}$$

6. 惩罚因子（Penalty Factor）

为了惩罚过多的词序差异，METEOR会计算一个惩罚因子，具体来说，惩罚因子会根据生成文本和参考文本之间的对齐方式来计算。

7. 综合计算

METEOR最终的得分通过以下公式综合计算：

$$\text{METEOR Score} = \text{Precision} \times \text{Recall} \times (1 - \text{Penalty})$$

METEOR得分公式：

METEOR的最终得分是通过精确度、召回率、同义词匹配等综合因素来计算的。METEOR的得分范围是0到1，得分越高，表示生成文本与参考文本的匹配程度越高。

METEOR的具体计算步骤：

1. **单词匹配**：生成文本和参考文本之间的精确匹配。
2. **同义词匹配**：在精确匹配的基础上，进一步考虑同义词的匹配。
3. **词形还原匹配**：通过词形还原（stemming）进一步增加匹配数量。
4. **计算精确度和召回率**：计算生成文本和参考文本中匹配单词的精确度和召回率。
5. **惩罚因子计算**：根据单词顺序的差异计算一个惩罚因子。
6. **计算METEOR得分**：综合精确度、召回率、惩罚因子来计算最终的METEOR得分。

举个例子

假设我们有以下参考文本和生成文本：

- **参考文本**（reference）：
  `"The cat sat on the mat."`
- **生成文本**（candidate）：
  `"A cat is sitting on a mat."`

1. 精确匹配：

- 参考文本和生成文本有两个完全匹配的单词：`"cat"` 和 `"mat"`。

2. 同义词匹配：

- 在这个例子中，没有涉及同义词的匹配，因此这一部分为零。

3. 词形还原匹配：

- `"sat"` 和 `"sitting"` 被视为匹配（通过词形还原）。

4. 计算精确度和召回率：

- **精确度 (Precision)**：匹配的单词数（3个）除以生成文本的单词数（6个），即 $\frac{3}{6} = 0.5$。
- **召回率 (Recall)**：匹配的单词数（3个）除以参考文本的单词数（5个），即 $\frac{3}{5} = 0.6$。

5. 惩罚因子：

- 由于生成文本和参考文本的单词顺序完全不同（`"A cat is sitting"` VS `"The cat sat"`），METEOR会计算一个小的惩罚因子。

6. 计算最终得分：

- 最终的METEOR得分会基于精确度、召回率和惩罚因子来计算，假设惩罚因子为0.1，则：

$$\text{METEOR Score} = 0.5 \times 0.6 \times (1 - 0.1) = 0.5 \times 0.6 \times 0.9 = 0.27$$

5. SPICE

https://blog.csdn.net/michaelshare/article/details/120939420

提取三元组命题，然后算命题匹配程度

- **参考文本**（Reference Caption）：

```css
A dog is jumping over a fence in the yard.
```

- **生成文本**（Generated Caption）：

```css
A dog is leaping over a fence in the yard.
```

步骤 1：命题提取

SPICE 将文本转化为 **命题**。每个命题由三个部分组成：

- **主体（subject）**：命题的主语
- **谓词（predicate）**：描述主语与宾语之间的动作或关系
- **宾语（object）**：描述动作或关系的目标

参考文本：

- `"A dog is jumping over a fence in the yard."`

从参考文本提取的命题：

1. `("dog", "is jumping over", "fence")`：表示“狗跳跃越过栅栏”
2. `("dog", "is in", "yard")`：表示“狗在院子里”

生成文本：

- `"A dog is leaping over a fence in the yard."`

从生成文本提取的命题：

1. `("dog", "is leaping over", "fence")`：表示“狗跳跃越过栅栏”
2. `("dog", "is in", "yard")`：表示“狗在院子里”

步骤 2：命题匹配

SPICE 比较生成文本和参考文本中的命题，计算它们的匹配情况。

参考文本的命题：

1. `("dog", "is jumping over", "fence")`
2. `("dog", "is in", "yard")`

生成文本的命题：

1. `("dog", "is leaping over", "fence")`
2. `("dog", "is in", "yard")`

命题匹配分析：

- 第一个命题：
  - 参考文本：`("dog", "is jumping over", "fence")`
  - 生成文本：`("dog", "is leaping over", "fence")`

  匹配情况：虽然两个命题在谓词上有细微差异，“jumping”与“leaping”是同义词，因此 SPICE 会视为 **语义匹配**。命题的主体和宾语完全一致。
- 第二个命题：
  - 参考文本：`("dog", "is in", "yard")`
  - 生成文本：`("dog", "is in", "yard")`

  匹配情况：完全匹配，主体、谓词、宾语都一致。

步骤 3：计算SPICE得分

1. 计算命题匹配率（Propositional Match Rate）

- **参考文本**总共有2个命题：`("dog", "is jumping over", "fence")` 和 `("dog", "is in", "yard")`。
- **生成文本**总共有2个命题：`("dog", "is leaping over", "fence")` 和 `("dog", "is in", "yard")`。

匹配命题的数量：

- 第一个命题（关于狗跳跃）是同义匹配，算作1个匹配。
- 第二个命题（关于狗在院子里）完全匹配，算作1个匹配。

匹配率为：

$$\text{Propositional Match Rate} = \frac{\text{匹配命题数}}{\text{参考文本命题总数}} = \frac{2}{2} = 1.0$$

2. 计算完整性（Completeness）

完整性衡量生成文本是否完全覆盖了参考文本的命题。在这个例子中，生成文本涵盖了参考文本的所有命题。

$$\text{Completeness} = 1.0$$

3. 计算惩罚因子（Penalty Factor）

SPICE 会考虑命题顺序的差异以及是否有不相关的命题。在这个例子中，生成文本与参考文本在命题顺序上完全一致，且没有缺失任何关键信息或添加无关命题。

因此，惩罚因子为：

$$\text{Penalty Factor} = 1.0$$

4. 计算最终SPICE得分

SPICE 的最终得分通过以下公式计算：

$$\text{SPICE Score} = \text{Propositional Match Rate} \times \text{Completeness} \times \text{Penalty Factor}$$

在我们的例子中：

$$\text{SPICE Score} = 1.0 \times 1.0 \times 1.0 = 1.0$$

6. Cider

https://blog.csdn.net/wl1710582732/article/details/84202254

7. Cider-D

加了一些去噪和权重调整

CIDEr-D与CIDEr的主要区别在于其更高的容错性，它在计算过程中通过减少对常见词汇和一些过于普遍的n-gram的依赖（识别重复n-gram），使得评价结果更加稳定和可靠，尤其在参考描述质量较差或多样性较高的情况下。