# 语句通顺性判断

## 一、 任务描述

判断给定的中文句子是否通顺，性能度量为不通顺作为正例时的 F1 值。

## 二、 技术路线

本次任务可以看作是句子分类，句子分类经典的模型是用 CNN 提取 N-gram 的特征，最后组合这些特征并将其投影到标签空间从而完成分类。经过实验，我发现该方案对于本任务来说效果并不好（F1 在验证集和 B 榜上都为 0.65 左右），当把 CNN 换位 RNN 时，模型的性能也并没有明显提升。

所以，我便尝试使用更为强大的预训练语言表征模型——BERT。BERT 在很多下游自然语言理解任务上都取得了 SOTA，而且本身很适合用来做句子分类任务。本次任务的实现基于 pytorch-pretrained-bert 库，使用的预训练中文模型来自 Google-Research。

## 三、 问题与解决

1. 问题：任务没有提供验证集。

   解决：要尽量保证验证集和测试集独立同分布，所以从训练集中挑选 1000 条正例和 1000 条负例作为验证集。

2. 问题：训练集正负样本数目严重不平衡，比例为 1:12.5。

   解决：若模型将正例误分为负例，则在原来的 loss 上乘以 12.5。

3. 问题：通过对模型在验证集上性能的观察，我发现模型识别正例的精度较高，将近 0.9，但召回率偏低且波动较大。

   解决：这说明模型只能学到部分不通顺句子的模式，还有相当的模式被忽略了，这个问题比较难以解决，值得进一步探究。但基于这个现象，自然地可以想到一种“免

费”提升性能的方法：使用保存的 checkpoint 给出预测，只要有一个 checkpoint 判断该句子不通顺，那么就认为该句子不通顺。经过实验，我发现这种策略可以使 F1 提升 3 个点。

## 四、 训练与测试

BERT-SMOOTH 以 2e-5 的学习率在训练集上 Fine-tune 1 轮，单 checkpoint 在验证集上取得 0.79 的 F1，在 B 榜上取得 0.78 的 F1。将 5 个 checkpoint 集成在 B 榜上取得 0.817 的 F1。

## 五、 程序说明

- 模型的训练，测试都在 run_smooth.py 中。
- run_smooth.sh 用于指定参数运行 run_smooth.py。
- preprocess.py 划分数据集。
- postprocess.py 对预测文件进行后处理。