文本数据增强常用方法及应用概述

HeYing

May 10th, 2019

文本增强主要是为了解决训练模型时文本数据稀缺的问题，由于它旨在解决由数据不足引起的过拟合问题，因此文本增强也可以看作是一种正则化技术。

目前已有的方法中，同义词替换时使用的最广泛的，另外还有通过注入噪声和拼写错误的方法、句法分析生成方法、正则表达式生成法、反向翻译法等。

本文主要从简易数据增强（包括同义词替换）、反向翻译、句法分析生成这三方面来阐述文本增强方法的思想及其实现、应用情况。

简易数据增强（Easy Data Augmentation, EDA）

简易数据增强是最近提出的文本数据增强概念（Wei & Zou, 2019），这一文本增强技术涵盖了之前常用的同义词替换方法（Kolomiyets et al., 2011; Zhang et al., 2015; Wang and Yang, 2015），并提出了新的三种文本数据增强操作：随机插入、随机交换以及随机删除。这四种简易方法主要是为了生成更多的尽量保留原语义的文本数据，最终目标是**用于训练出更稳健的模型**。

同义词替换（Synonym Replacement, SR）

在生成文本研究中，同义词替换是一个很自然的想法，即：给定一个句子，将句中的一些词语用其同义词替代，从而产生新的但语义不变的句子。在带标签数据稀缺的情况下，这一方法在许多研究者的实验中被证明是有效的（Simard et al., 2003; Kolomiyets et al., 2011; Zhang et al., 2015; Wang and Yang, 2015）。

同义词替换在实验操作时经常面临的问题主要有以下三个：

1. 哪些词应该被选为 “待替换词”
2. 应该选择众多同义词中的哪个词来执行替换
3. 给定一个句子，应替换掉多少个词来产生新的语句

这里需要注意的一点是，执行替换之后产生的新语句是否符合语法逻辑并非评估文本生成效果的重点，因为最终目的是服务于建立更稳健的模型，因此生成文本数据的评估重点应在于新句子的语义是否与原语句的标签信息相符合。

尽管有相关的研究给出了一些执行替换的建议，比如仅替换形容词和副词、少替换动词、限制不能被替换的词语等等，但这些建议在保证生成文本尽可能保持语义不变的同时，也带来了比较大的缺陷——生成文本的形式、结构都太过相似了，过于相似的样本解决模型训练时过拟合问题的能力就会有所下降，因

此，近期的应用研究着力于克服这一缺陷，一种可行的操作方法是：在给定的每个句子中，随机选取 `n` 个非停用词，分别随机地选取它们的一个同义词替换它们。

关于如何获取同义词，主要的方法有 2 个：

1. 使用公开的的同义词词库，如WordNet、常用的中文近义词工具包Synonyms、哈工大刘挺教授团队研发的大词林等；
2. 使用大规模语料库训练语言表征模型，或者下载开源的预训练好的模型，得到各个词语相应的词向量之后，计算它们之间的相似度从而找到其同义词。

随机插入（Random Insertion, RI）

随机插入操作是在Wei & Zou, 2019的研究中首次被提出的（下文的随机交换与随机删除同），具体做法是：给定一个句子，随机选择其中的一个非停用词，然后随机选择该词的一个同义词，将这个同义词插入到原句中的随机一个位置上，重复这个过程 `n` 次。

这个想法听起来很新奇，虽然大部分情况下随机插入后生成的句子在语法上不是通顺的，但由于随机插入的是同义词，同义词对于原文本的标签信息一般不会产生相反的作用，而充分的实验也证明这一操作在 RNN 类和 CNN 类模型的分类任务中都取得了比较显著的性能提升。

随机交换（Random Swap, RS）

随机交换的具体操作是：给定一个句子，随机选中其中两个词语，交换他们的位置，重复这个过程 `n` 次。

交换之后很可能会改变语义，比如在句子 `Why do cats like mice?` 中，交换 `cats` 和 `mice` 的位置得到的句子语义是很不同的，所以 Wei 等人的研究中提到，执行这一操作时，对于一个句子，交换次数 `n` 要控制好，在他们的实验中，`n` 的取值为句子总长的 `20%` 以下时，这一操作对模型训练的性能提升比较可观；而当 `n` 过大时，过多的词语位置交换相当于打乱了原始句子所蕴含的语序信息。

随机删除（Random Deletion）

随机删除的具体操作是：给定一个句子，句中的每个词语以概率 `p` 被随机删除。

与随机交换操作类似地，当随机删除概率 `p` 较小时，对于模型训练的性能提升是有效的，但当概率 `p` 过大（比如 `p = 0.5`，即一半的词语被随机删去）时，性能反而会下降，这是因为删去过多的词语会导致新生成句子的整体性和连贯性受损，句子变得难以理解。

EDA 性能表现评估

对以上 4 中简易文本数据增强方法的评估，主要从原语义保留、模型性能提升、操作成本这三方面来总结。

原语义保留

经过 EDA 操作之后产生的新文本数据，是否较好地保留了原来句子的语义信息是评价其数据生成质量的重要指标，而“语义保留”这一概念其实是比较抽象的，难以用一个具体的数值统计量来描述，因此实验研究中一般通过判断生成句子的信息是否符合原句子的标签信息来判断其语义保留情况，为了能更形象直观地呈现这种“标签信息保留”情况，可以采用 t-SNE (Van Der Maaten, 2014) 方法来进行

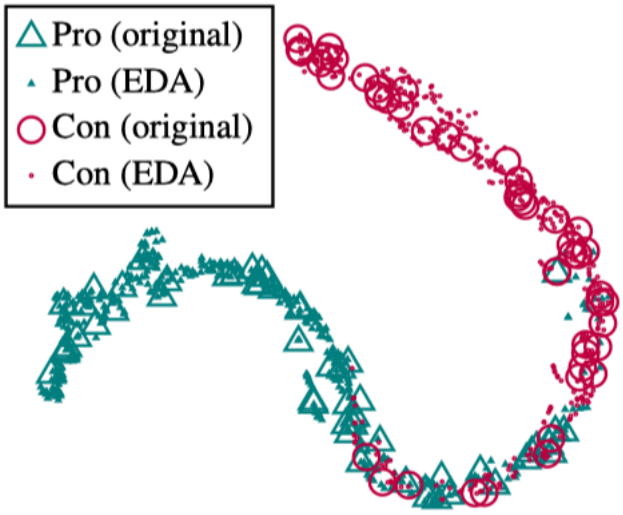


Figure 2: Latent space visualization of original and augmented sentences in the Pro-Con dataset.

图 1: 使用 t-SNE 可视化增强的 Pro-Con 数据集

可视化，t-SNE 算法可以简单地理解为一种复杂的非线性映射，其作用是将高维的文本嵌入向量映射到低维空间中，语义信息越相似的文本在这个低维空间中的距离就越近。通过 t-SNE 对 Pro-Con 数据集 (Ganapathibhotla and Liu, 2008) 中的样本进行映射，原始文本数据和生成文本数据在 2 维空间中的分布如图 1，从图中可以看到，在潜在语义的低维空间中，EDA 生成的句子围绕在原始句子周围，由此可知生成句子基本保留了原始语义信息。

模型性能提升

在 Wei 等人的实验研究中，使用了 5 个流行的文本分类任务数据集：(1) SST-2: 斯坦福情感分类数据集 Treebank (Socher et al., 2013); (2) CR: 电影用户评论 (Hu and Liu, 2004; Liu et al., 2015); (3) SUBJ: 主客观数据集 (Pang and Lee, 2004); (4) TREC: 问题匹配数据集 (Li and Roth, 2002); (5)PC: 支持-反对数据集 (Ganapathibhotla and Liu, 2008)。他们将这 4 中 EDA 方法组合应用到这 5 个数据集对应的分类任务上，使用了 RNN 和 CNN 进行模型训练，得到各类算法在不同数据集任务上的平均表现，将其与未执行 EDA 操作的模型结果进行比较，如图 2 所示。

从上图曲线可以看到，在使用的训练集数据比例较小时，也就是数据量稀缺的情况下，EDA 方法的应用一定程度上提高了模型的性能。

图 3 则进一步展示了有无 EDA 操作时在具体的不同算法上性能提升的汇总。

操作成本

通过和两个其他研究方法的成本进行比较来评估 EDA 的操作成本是否具有优势，一个研究是 Hu(2017) 使用 VAE（variational autoencoder，变分自动编码器）框架来生成文本数据，另一个是 Kobayashi (2018) 使用双向语言模型生成文本数据，三者的成本与性能提升汇总如图 4 所示。

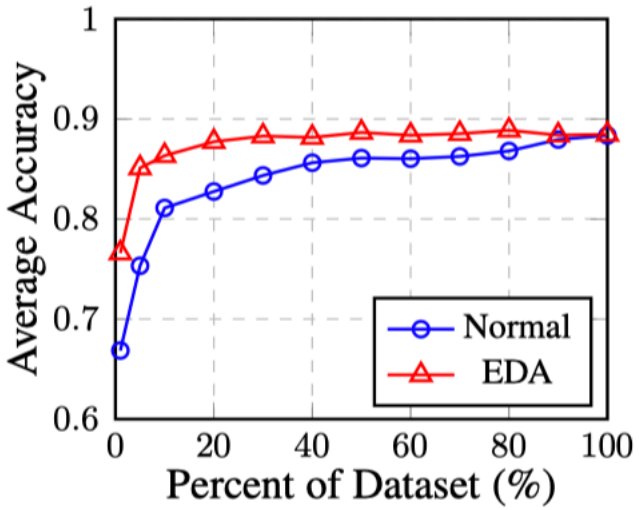


Figure 1: Performance on text classification tasks with respect to percent of dataset used for training.

图 2: EDA 文本增强在分类任务的表现对比

	Training Set Size			
Model	500	2,000	5,000	full set
RNN	75.3	83.7	86.1	87.4
+EDA	79.1	84.4	87.3	88.3
CNN	78.6	85.6	87.7	88.3
+EDA	80.7	86.4	88.3	88.8
Average	76.9	84.6	86.9	87.8
+EDA	79.9	85.4	87.8	**88.6**

Table 2: Average performances (%) across five text classification tasks for models with and without eda on different training set sizes.

图 3: EDA 增强方法在不同算法上的性能提升

Technique (#datasets)	Gain	LM	Ex Dat
VAE+discrim.[2] (2)	∼3%	yes	yes
Contextual aug.[3] (5)	0.5%	yes	no
EDA (ours) (5)	**0.8%**	**no**	**no**

Table 4: Related work in data augmentation. LM: requires training a language model or deep learning. Ex Dat: requires an external dataset. #datasets: number of datasets used for evaluation.

图 4: EDA 方法与其他相关方法的成本及性能提高对比

从图 4 中可以看到 EDA 方法与其他两个方法相比，没有使用语言模型（Language Model, LM），也没有使用额外的数据信息，采用 VAE 框架的文本生方法性能提升达到了 3%，但同时使用到了语言模型和额外数据集，由于它需要训练深度学习网络，因此消耗的计算资源成本以及时间成本都较高；与 Kobayashi 的方法相比，EDA 方法生成时没有使用语言模型，但性能提升要更高一些。

尽管 EDA 方法的性能提升甚微，但 EDA 方法的简易性为它在 NLP 中的广泛使用提供了一个令人难以抗拒的理由。

EDA 实现

Wei 和 Zou 开源了他们论文里阐述的EDA方法，这一开源项目主要适用于英文文本的生成。

Zhan 在原 EDA 项目的基础上，推出了一个中文实现版本，还有另一个版本在此基础上加了一些过滤，同时调高了同义词替换、同义词插入的权重，项目详情见这里。

反向翻译

反向翻译（Back-Translation, 又称回译）是近年来很流行的一种文本数据增强方法。它的思想十分简单，通过将原始句子翻译到另外一种语言，再翻译回原始语言，从而生成新的语义相同但表达略有不同的语句，达到文本增强的目的。

反向翻译的想法源于机器翻译领域，机器翻译依赖于大型平行语料库，即源语和目的语中成对句子的数据集，但现实情况中，双语语料往往是十分有限的，因此很自然地想到可以通过反向翻译来增强双语数据。

关于反向翻译的理论研究近年主要都是聚焦在训练机器翻译模型的性能提升这方面，其中最具代表性的是 Google 和 Facebook 的合作成果《Understanding Back-Translation at Scale》(Edunov et al., 2018)，这一研究扩展了之前有关于反向翻译方法的分析（Sennrich et al. , 2016a ; Poncelas et al. , 2018），基于 WMT 竞赛公共双语语料，向该双语语料中添加了数亿个反向翻译得到的句子，对神经机器翻译的反向翻译进行了大规模的研究，通过实验证明了基于采样或者添加噪声的 beam search（集束搜索）比单纯的 beam search 在多个不同的基准数据集上的 BLEU 得分平均要高 1.7 个百分比。这篇论文发表于 EMNLP 2018，同时作者们开源了这个项目。

相关应用实现

实现反向翻译可以从两方面考虑：

- 一是基于大料的语料库训练自定义的机器翻译模型，再使用训练好的机器翻译模型实现反向翻译。这一方法要求我们要有足够量的语料数据，而且最好是多种语言的语料数据，因为基于多语言训练好的机器翻译模型有助于更好地实现文本增强；但现实情况中，个人所能获取的或者公开的语料数据往往是稀缺的，因此训练出来的机器模型性能受到了限制。
- 二是使用现有的翻译工具，如谷歌、有道、百度、腾讯、搜狗等企业均有其自行研发的在线翻译工具，通过调用这些翻译工具来实现反向翻译。

这里主要总结第二种方法的反向翻译实现。

调用在线翻译工具 API 实现反向翻译

不少在线翻译工具提供了 API 借口，也有相应的开源模块可以供用户通过 python 实现 API 的调用，但使用企业提供的 API 有一个局限就是需要按字符数进行付费，总结了以下在线翻译工具的支持语言情况以及收费标准：

- 谷歌翻译：支持 128 种语言，是支持语言数目最多的翻译工具，50 万字符以内免费，超过 50 万字符按 20 美元/百万字符收取费用；
- 有道翻译：支持 107 种语言，注册后账户里有 100 元体验金，48 元/百万字符，价格比较昂贵；
- 百度翻译：支持 28 种语言互译，注册账户后每月有 200 万字符的流量，超出则按 49 元/百万字符收费；
- 腾讯君翻译：支持 15 种语言互译，目前价格未知，有请求次数的限制；
- 搜狗翻译：支持 78 种语言，注册后账户有 200 元体验金，常见语言按 40 元/百万字符收费, 非常见语言则是 60 元/百万字符。

可以通过查看各翻译工具官网提供的 API 调用文档来实现反向翻译。

动态爬虫实现无字符限制的在线翻译

从上一小节的各翻译工具收费总结可以看到，相比起我们所需要的语料数据的数量，免费翻译的字符额度是十分有限的，如果个人使用纯粹的 API 调用，则价格成本很高，因此需要通过别的方式来降低语料获取的成本。

通过程序自动获取在线翻译的结果本质上是一种爬虫技巧：我们在任何一个翻译网站的左侧内容栏中输入想要翻译的语句，然后点击翻译按钮，于是在网页右边的内容栏里会出现翻译结果，所以我们的程序所要做的是向翻译网站发送带翻译语句，然后模拟点击翻译按钮，再将翻译结果语句爬取下来。

每个翻译工具的翻译解析具体方法都有所不同，但一般都是找到其时间戳的加密方法，此外还要构造好 cookie，设置好用户代理等，这些对于应对商业网站的反爬虫策略是十分必要的。

这里列出几个关于 Python 的在线翻译实现：

- 谷歌翻译、腾讯君反向翻译

- 百度翻译实现
- 我自己实现并经过测试的有道翻译、谷歌翻译

以上列出的实现中，后面 2 个没有一步到位实现反向翻译，但通过调用两次，分别指定翻译语言从语言 A 到语言 B 的参数（参数 from 和参数 to），即可实现反向翻译。

翻译结果的质量评估

尽管网络上有不少开源的在线翻译实现能够用于文本增强任务，但是反向翻译得到的语句质量是否符合与其要求却鲜有人进行测试和对比分析。

关于生成文本的语义是否与原始文本保持一致，除了前文介绍 EDA 方法时提到的 t-SNE 可视化方法，还有其他的一些语义相似度衡量的方法，比如最新的 NIPS 2018 有一篇论文《Learning semantic similarity in a continuous space》提出了一种基于 VAE 框架的连续空间中的语义相似度衡量方法。近年来关于语义相似度衡量的方法大部分都基于深度学习模型，这类方法的特点是准确率高（目前 SOA 水平接近 90%），但是缺点在于模型构建成本过高。

实际上，由于不同的在线翻译工具使用的机器翻译算法和训练模型的语料不同，使得它们在对不同语言的翻译上优势不同，比如一些经验会建议你，阅读学术论文时，使用谷歌翻译的效果是最佳的，因为它能对论文里的复杂长句有更准确的解析和翻译结果，再比如，在翻译一些短文本或者日常生活用语时，经验告诉你有道翻译的效果会更好（这里的举例基于英译中场景），等等。

因此，一个比较直观简单的想法是，通过经验总结出不同的翻译工具在不同语言翻译上的表现，通过充分的实验进行验证之后，作为经验值来使用。

以经典的问题匹配数据集quora_duplicate_questions.tsv为例，我分别使用有道翻译和谷歌翻译对这个英文文本进行了翻译得到中文文本，通过观察发现有道翻译的翻译结果比谷歌翻译要好不少，详细的翻译结果对比举例可见项目内容。

在生成所需要的数据时，可以先在小数据集上比较各种工具的翻译表现，从而选择效果最佳的在线翻译工具来生成新数据，如此得到的文本数据质量也会更高。

句法分析生成

句法分析即对句子中各成分之间的相关依赖关系进行分析，例如谓语的直接发出者和作用对象、形容词修饰的名词词组等句子成分之间的关系，表示句法关系的一个有效工具是句法依赖树（Gagnon & Da Sylva, 2005; Zouaq et al., 2010），在句法树中，节点为给定句子的各个词（组），边为各词（组）之间的具体依赖关系。

通过句法分析来生成新的文本数据的框架大致如图 5 所示。

从图 5 的分析流程图可知主要有三个步骤：

1. 将文本通过句法分析器转化为句法依赖树；
2. 对句法依赖树进行转换，生成结构不同但蕴含相同语义的新句法依赖树；
3. 将新生成的句法依赖树转换成文本，得到增强文本数据。

接下来将从这三方面进行方法应用上的阐述。

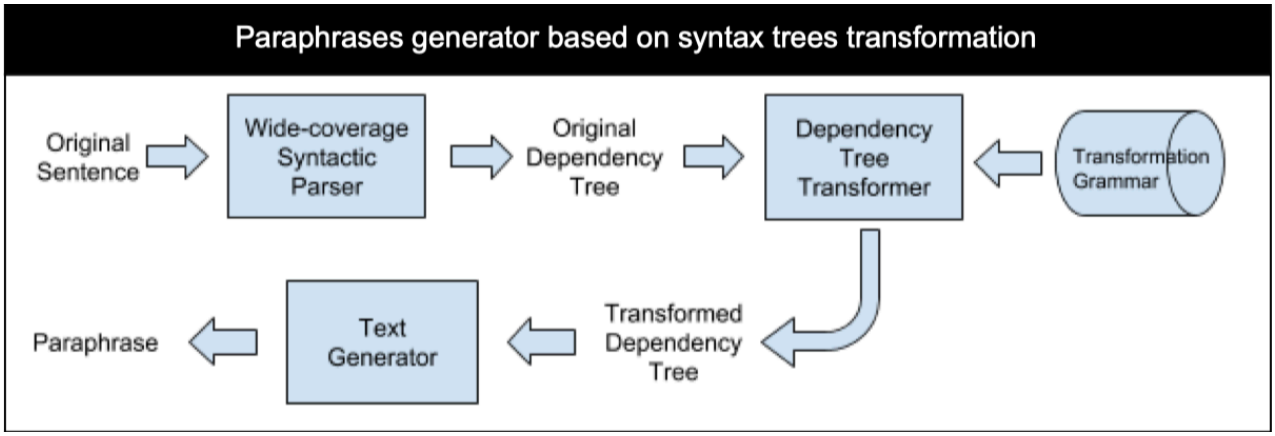


图 5: 句法分析实现文本增强的方法流程

文本转换成句法依赖树

关于句法树生成的相关研究比较丰富，主要分为以下几类：基于规则、基于统计和基于深度学习的方法，其中基于统计和深度学习的方法是近年的主流。

一个句法树生成器的得到需要有训练数据集以支撑模型训练，在句法分析领域，经典的标记数据集有Penn Treebank，中文语义依存图数据集SemEval-2016等，除此之外 CoNLL 也经常开放句法分析的相关评测及对应特定任务的数据集。

到目前为止已经有许多开源的句法分析工具可供使用来直接得到句法分析树，而不用从头开始训练一个句法生成器，除了著名的谷歌开源句法分析工具SyntaxNet之外，还有以下几个流行的实用句法分析工具：

- StanfordCoreNLP：斯坦福开源的 NLP 项目，包含句法分析功能；
- HanLP：HanLP 是一系列模型与算法组成的 NLP 工具包，其中提供了中文依存句法分析功能；
- FudanNLP：复旦大学自然语言处理实验室开发的中文自然语言处理工具包，含有句法分析功能。

通过这些实用工具，我们可以实现将文本转化为句法树，用于后续的文本增强流程。

生成新的句法依赖树

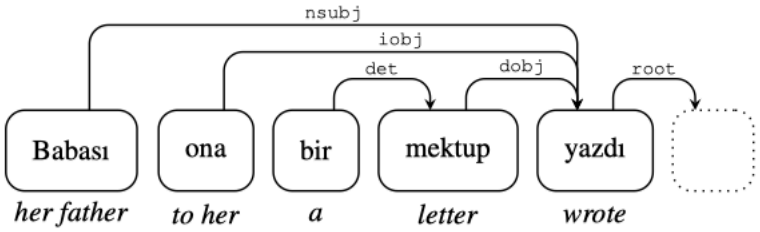
在得到原语句的句法分析树后，接下来的步骤就是对这一句法树进行转换，目的是生成新的结构不同但包含相似语义的句法树，这一步是整个句法分析生成增强文本的关键一步。

在 Coulombe 的实验研究《Text Data Augmentation Made Simple By Leveraging NLP Cloud APIs》中提到，这一语法树转换的过程是根据一系列专家人工构造的语法规则来进行的。

另外，最近有新的研究借鉴了图像生成领域 (Image Generation) 的思想（Sahin & Steedman, 2018），对原文本的句法树采取了“修剪”（crop）和“旋转”（rotate）操作，来生成新的句法树。

如图 6 所示，是 Sahin 和 Steedman 的研究中对于句法树修建和旋转的一个示例，该示例为一个土耳其语句 “Babası ona bir mektup yazdı”，英文为 “Her father wrote her a letter”，具体的操作方法为：

- **修剪**：图像领域的修剪是将图像外围除去，一般是为了更集中地突出图像中的主体，比如突出一幅图中一片草地上的一朵花，句法分析树的修剪借鉴了这一思想，先辨别出句子中关键的词组，再将



(a) Dependency analysis

(1) Babası yazdı *(Her father he-wrote)*
(2) Ona yazdı *(He-wrote to her)*
(3) Bir mektup yazdı *(He-wrote a letter)*

(b) Sentence Cropping

(1) Babası yazdı bir mektup ona (SVOIO)
(2) Yazdı babası ona bir mektup (VSIOO)
(3) Bir mektup yazdı babası ona (OVSIO)
(4) Ona bir mektup yazdı babası (IOOVS)

(c) Sentence Rotating

Figure 1: Demonstration of augmentation ideas on the Turkish sentence "Babası ona bir mektup yazdı" *(Her father wrote her a letter)*. S: Subject, V: Verb, O:Object, IO: Indirect Object. Arrows are drawn from dependent to head. Both methods are applied to the *Labels of Interest (LOI)*.

图 6: 文本增强的“修剪”和“旋转”操作示例

				crop			rotate			
#Tokens	**Lang**	**Type**	**Org**	$p = 0.3$	$p = 0.7$	$p = 1$	$p = 0.3$	$p = 0.7$	$p = 1$	*Imp%*
$< 20K$	Lithuanian	*IE, Baltic*	61.51	62.17	66.28	67.64	65.28	66.56	**68.27**	*10.98*
	Belarusian	*IE, Slavic*	83.58	83.87	85.50	85.39	84.33	85.96	**86.11**	*3.03*
	Tamil	*Dravidian*	81.93	81.35	82.78	**84.34**	83.74	83.86	83.61	*2.94*
	Telugu	*Dravidian*	90.78	**90.85**	89.88	90.50	90.36	90.29	89.95	*0.07*
	Coptic	*Egyptian*	**95.17**	94.60	94.74	94.12	95.03	94.65	94.60	*-0.15*
$< 80K$	Irish	*IE, Celtic*	62.75	73.72	75.87	75.42	72.51	**76.35**	76.19	*21.68*
	North Sami	*Uralic, Sami*	86.78	86.04	87.17	87.35	87.85	**88.04**	86.65	*1.45*
	Hungarian	*Uralic, Ugric*	85.94	86.24	86.56	**86.62**	86.49	86.37	86.60	*0.80*
	Vietnamese	*Austro-Asiatic*	75.16	**75.59**	75.32	74.84	75.22	75.15	75.14	*0.57*
	Turkish	*Turkic*	93.49	93.53	93.56	93.89	93.60	93.82	**93.98**	*0.52*
	Greek	*IE, Greek*	95.18	95.32	95.46	**95.54**	95.26	95.22	95.35	*0.38*
	Gothic	*IE, Germanic*	94.38	94.42	94.35	94.44	**94.62**	94.48	94.43	*0.25*
	Old Slavic	*IE, Slavic*	95.36	95.34	95.33	**95.44**	95.17	95.35	94.93	*0.08*
	Afrikaans	*IE, Germanic*	94.91	94.52	94.86	**94.93**	94.73	94.70	94.92	*0.0*
$< 120K$	Latvian	*IE, Baltic*	91.22	91.38	91.77	**91.78**	91.69	91.62	91.76	*0.61*
	Danish	*IE, Germanic*	94.25	94.17	93.96	**94.78**	94.18	94.10	94.21	*0.56*
	Slovak	*IE, Slavic*	91.23	91.17	91.04	91.35	91.53	91.38	**91.58**	*0.38*
	Serbian	*IE, Slavic*	96.14	96.26	96.12	96.17	**96.35**	96.16	96.07	*0.22*
	Ukranian	*IE, Slavic*	94.41	94.33	94.56	94.49	**94.57**	94.38	94.47	*0.17*

Table 1: POS tagging accuracies on UDv2.1 test sets. Best scores are shown with **bold**. Org: Original. p: operation probability. *Imp%*: Improvement over original (Org) by the best model trained with the augmented data.

图 7: 加入经修剪和旋转增强文本数据的模型性能表现

其他不重要的模块除去。如上图的中间部分所示，将句子的重心放在主语（her father）、非直接宾语 (to her) 以及对应的谓语 (wrote) 上，并将其他非核心元素以及它们的边去掉，从而构建出 3 个新的短句。尽管得到的新短句在语法上可能会稍不顺畅，但基本保留了原有的语义，甚至使得语义更加浅显易懂。

- **旋转**：图像处理中，旋转是指将图像绕其中心旋转一定的角度，从而得到新的图像；在句法分析增强中，借鉴旋转的思想，首先选择句法树的根部作为旋转中心，然后将根部附近的可移植碎片进行旋转，从而得到新的句法树。可移植碎片通常由语言的形态分类学来定义（Futrell et al., 2015）。如上图底部所示，以谓语（wrote）为树的根部，旋转其周围的可移植碎片，可以得到新的句法树。需要指出的是，旋转后的句法树是否符合语言逻辑和语言本身有关，分析类语言比如英语，进行旋转很可能会打破原来语言的内在语义顺序，因此通过旋转很可能会引入噪声；而像拉丁语、土耳其语、希腊语等语言，它们对句子中元素的顺序要求很低，经过旋转得到的句子语义基本能够保持一致，因此这类语言更适合通过旋转来增强文本数据。

Sahin 和 Steedman 用上述句法树修剪和旋转得到的增强文本数据训练一个双向 LSTM 模型用于一个字符级别的序列标注任务，结果表明修剪和旋转操作对于模型性能有普遍的一定程度上的提高，如图 7 所示。

相应的代码也已开源。

这些操作方法在中文领域是否适用还有待进一步的实验探索。

句法依赖树转换为文本数据

生成新的句法树之后，还需要把句法树转化为文本，从而完成文本数据增强的流程。

python 中的 **nltk.corpus** 模块的 **treebank_chunk** 提供了这一功能，它的思想是把树的节点以及属性提取出来，通过内在的语法规则将其重新组合成一个文本句子。

一个简单的例子如下：

对于一棵语法树，

```
(S
(NP Pierre/NNP Vinken/NNP), /,
(NP 61/CD years/NNS)
old/JJ, /,
will/MD
join/VB
(NP the/DT board/NN)
as/IN
(NP a/DT nonexecutive/JJ director/NN Nov./NNP 29/CD)
./.)
```

提取出节点及其属性如下，

```
[('Pierre', 'NNP'), ('Vinken', 'NNP'), (', ', ', '), ('61', 'CD'),
('years', 'NNS'), ('old', 'JJ'), (', ', ', '), ('will', 'MD'), ('join', 'VB'),
('the', 'DT'), ('board', 'NN'), ('as', 'IN'), ('a', 'DT'), ('nonexecutive', 'JJ'),
('director', 'NN'), ('Nov.', 'NNP'), ('29', 'CD'), ('.', '.')]
```

生成文本语句：

```
Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29 .
```

具体用法可参考这里。

参考文献

Jason Wei, Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. arXiv:1901.11196. online

Oleksandr Kolomiyets, Steven Bethard, and MarieFrancine Moens. 2011. Model-portability experiments for textual temporal analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2, HLT '11, pages 271–276, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, pages 649–657, Cambridge, MA, USA. MIT Press.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using

#petpeeve tweets. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 2557–2563.Association for Computational Linguistics.

Patrice Simard, Yann LeCun, John S. Denker, and Bernard Victorri. 1998. Transformation invariance in pattern recognition-tangent distance and tangent propagation. In Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop, pages 239–27, London, UK, UK. Springer-Verlag.

Laurens Van Der Maaten. 2014. Accelerating t-sne using tree-based algorithms. J. Mach. Learn. Res., 15(1):3221–3245.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1, COLING '08, pages 241–248, Stroudsburg, PA, USA. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, pages 168–177, New York, NY, USA. ACM.

Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, pages 1291–1297. AAAI Press.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In Proceedings of the 19th International Conference on Computational Linguistics - Volume 1, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1, COLING '08, pages 241–248, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In ICML.

Sosuke Kobayashi. 2018. Contextual augmentation:Data augmentation by words with paradigmatic relations. In NAACL-HLT.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. Conference of the Association for Computational Linguistics (ACL).

Alberto Poncelas, Dimitar Sht. Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. 2018. Investigating backtranslation in neural machine translation. arXiv, 1804.06189.

Michel Deudon. Learning semantic similarity in a continuous space. Advances in Neural Information Processing Systems 31 (NIPS 2018).

Gagnon, M., & Da Sylva, L. 2005. Text summarization by sentence extraction and syntactic pruning.

Zouaq, A., Gagnon, M., & Ozell, B. 2010. Semantic analysis using dependency-based grammars

and upper-level ontologies. International Journal of Computational Linguistics and Applications, 1(1-2), 85–101.

Claude COULOMBE. 2018. Text Data Augmentation Made Simple By Leveraging NLP Cloud APIs. arXiv:1812.04718. online

Gozde Gul Sahin, Mark Steedman. 2018. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 5004–5009. Association for Computational Linguistics.

Richard Futrell, Kyle Mahowald, and Edward Gibson. 2015. Quantifying word order freedom in dependency corpora. In Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015), pages 91–100. Uppsala University, Uppsala, Sweden.