Enhancing Aspect Term Extraction with Soft Prototypes

Zhuang Chen, Tieyun Qian

摘要

方面词抽取任务（Aspect Term Extraction，ATE）是方面级情感分析中的一个基础性子任务。给定一个评论文本，ATE 的目标是抽取被用户表达了情感的方面短语。由于缺少包含尾部词的样本，序列标注器可能会收敛到较差的状态。本文提出基于软模板的 SoftProto 框架来增强方面词抽取任务，旨在解决评论文本中方面词和环境词具有长尾分布的问题。SoftProto 框架几乎可以与所有的序列标注器进行结合。在多个 SemEval 数据集上的实验表明，软模板的引入大幅度地提升了几个经典序列标注器在方面词抽取任务上的性能。

关键词：情感分析；方面词；SoftProto；序列标注器

1 引言

ATE 的目标是抽取被用户表达了情感的方面短语。例如对于评论“The Bombay style bhelpuri is very palatable.”，ATE 希望抽取出方面词“bhelpuri”。ATE 在过去二十年间已被广泛研究。早期的研究多致力于设计规则或是手工特征实现抽取。随着深度学习的发展，目前多数研究都将 ATE 当作一个序列标注任务，并设计序列标注器为评论生成对应的标签序列。虽然现有的序列标注方法在 ATE 任务上已经取得了优良的性能，但它们仍然面对一个严峻的挑战：由于缺少包含尾部词的样本，序列标注器可能会收敛到较差的状态。如图 1所示，在常用的 SemEval 数据集中，大约有 80% 的方面词和环境词（即非方面词）都出现不超过 5 次。根据相关研究，在训练样本不足的情况下，神经网络模型很难收敛到最优状态。

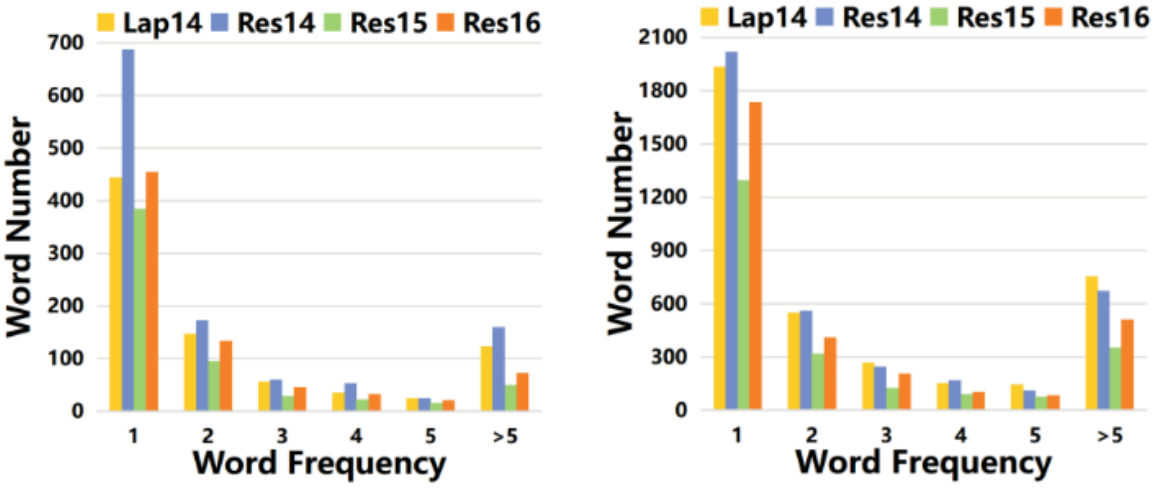


图 1: SemEval 数据集中方面词（左图）与环境词（右图）的分布

为了解决上述问题，我们的基本设想是将样本相互关联起来，从而帮助罕见词的抽取。例如，如果我们将前例中的罕见方面词“bhelpuri”与常见方面词“food”关联起来，与“bhelpuri”相关的样本就会变得很丰富。

2 相关工作

为了建立这种关联，寻找同义词是一个直观的想法，但该方法存在两个问题：首先，词典中只有小部分词能找到确定的同义词，虽然可以采用词向量寻找最近邻，但其语义相似性并不能得到保证；其次，方面词的存在是动态的，需要根据是否有针对该词的观点来确定。因此，我们需要建立一种动态的关联关系，且要从单词的上下文而非单词本身入手。[1]。

2.1 软检索方法建立单词级的关联

如图 2所示，在进行软检索之后，我们可以得到一个生成的样本，其与原样本在词级一一对应。我们将其称为“软模板”，因为其可以作为一个参考点来指导模型对于原样本的学习过程。

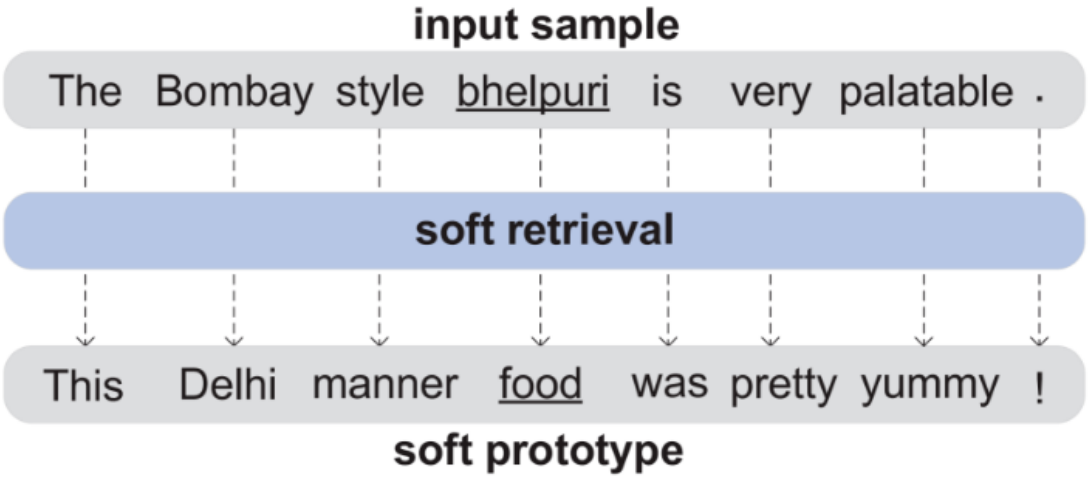


图 2: 软检索过程示意图

2.2 语言模型 LM 实现软检索

作为一个自监督的任务，语言模型的建模过程不需要额外标注，且能吸收领域内的全局知识。此外，现有研究表明，语言模型倾向于生成常见的输出，这恰好满足了我们将罕见词与常见词关联起来的需求。

2.3 语料预训练

具体地，我们首先根据给定的语料预训练双向语言模型（语料可以来自训练集或外部无标注数据），接着固定语言模型，再根据单词的上下文来推断其对应的模板词。我们将生成的软模板当作标注方面词的辅助证据，从而为模型判别长尾词提供助力。

3 本文方法

如图 3所示，SoftProto 框架由三部分组成：

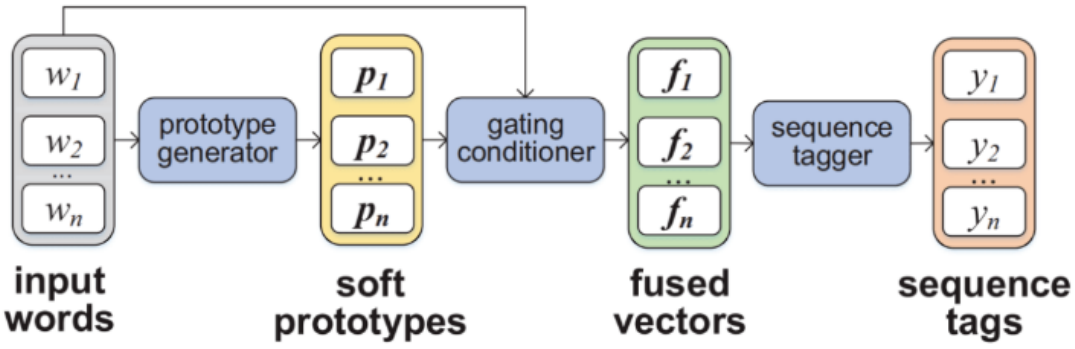


图 3: SoftProto 框架

3.1 模板生成器

用于实现软检索过程，并为样本生成对应的软模板。模板生成器的工作过程分为两部分，如图 4所示。首先根据给定的语料预训练双向的语言模型。随后，固定语言模型的参数，就可以根据某一位置的前文或后文推断该处可能的词。语言模型在某一位置的输出为一个词表大小的概率分布，我们取出 top-K 个候选词（本文称作 Oracle Words），并按照其对应的概率，对词向量进行加权求和，可计算出该位置上前向或后向的软模板向量，最终软模板向量取前后向模板向量的均值。将每一位置的软模板向量按顺序排列，即可获得样本对应的软模板序列。

3.2 门控调制器

用于融合样本与软模板的知识，并生成融合后表示。门控调制器通过对样本表示和软模板表示进行两方面的操作来促进融合：第一，软模板自身包含了可以作为支撑证据的信息，因此先将每个单词的向量与其对应的软模板向量进行拼接；第二，软模板向量可以提纯原样本的表示，因此再对拼接向量的每一维做门控操作，最终可获得融合后向量。如下式，其中 x 为原样本中的单词向量，p 为对应的模板词向量，f 为融合后向量。

$$\boldsymbol{f_i} = \sigma(W(\boldsymbol{x}_i \oplus \boldsymbol{p}_i) + b) \odot (\boldsymbol{x}_i \oplus \boldsymbol{p}_i)$$

3.3 序列标注器

序列标注器的目标是从融合后向量中提出高层语义特征，并据此预测标签序列。由于软模板独立于序列标注器，因此可以选用任意现存的标注器作为基准。根据标注器的预测结果，与真实标签计算交叉熵损失，即可端到端地训练 SoftProto 框架（语言模型的预训练不包含在训练过程中）。

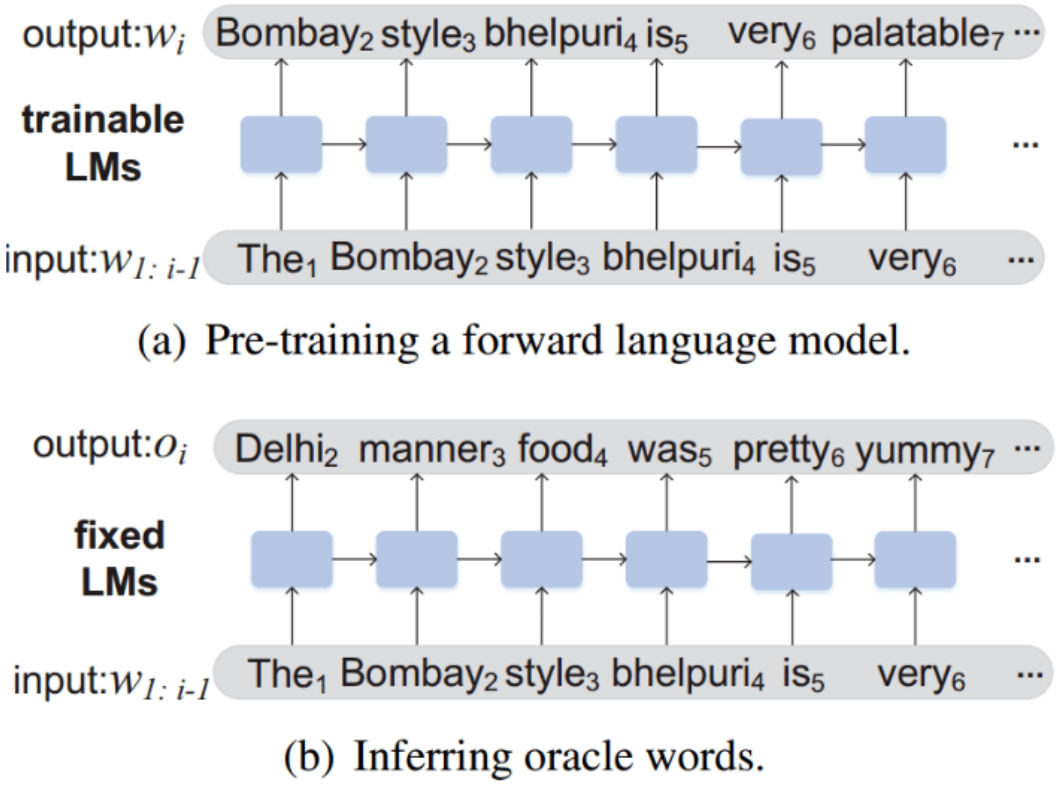


图 4: 语言模型的预训练与推断过程

4 复现细节

4.1 与已有开源代码对比

---

python train$_s oftproto.py --datasetlap14 --lmNone --seed123$
$pythontrain_s oftproto.py --datasetlap14 --lmNone --seed321$
$pythontrain_s oftproto.py --datasetlap14 --lmNone --seed111$
$pythontrain_s oftproto.py --datasetlap14 --lmNone --seed222$
$pythontrain_s oftproto.py --datasetlap14 --lmNone --seed333$

python train$_s oftproto.py --datasetlap14 --lminternal --seed123$
$pythontrain_s oftproto.py --datasetlap14 --lminternal --seed321$
$pythontrain_s oftproto.py --datasetlap14 --lminternal --seed111$
$pythontrain_s oftproto.py --datasetlap14 --lminternal --seed222$
$pythontrain_s oftproto.py --datasetlap14 --lminternal --seed333$

python train$_s oftproto.py --datasetlap14 --lmexternal --seed123$
$pythontrain_s oftproto.py --datasetlap14 --lmexternal --seed321$
$pythontrain_s oftproto.py --datasetlap14 --lmexternal --seed111$
$pythontrain_s oftproto.py --datasetlap14 --lmexternal --seed222$
$pythontrain_s oftproto.py --datasetlap14 --lmexternal --seed333$

对 SoftProto 的训练和评估。

4.2 实验环境搭建

python 3.6.5

pytorch 1.5.0

pytorch-pretrained-bert 0.4.0

numpy 1.19.1

```
(base) C:\Users\max>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

```
(base) C:\Users\max>pip install pytorch-pretrained-bert
Collecting pytorch-pretrained-bert
  Downloading https://files.pythonhosted.org/packages/d7/e0/c08d5553b89973d9a240605b9c12404bcf8227590de62bae27acbcfe076b
/pytorch_pretrained_bert-0.6.2-py3-none-any.whl (123kB)
    100% |████████████████████████████████| 133kB 167kB/s
Collecting tqdm (from pytorch-pretrained-bert)
  Downloading https://files.pythonhosted.org/packages/47/bb/849011636c4da2e44f1253cd927cfb20ada4374d8b3a4e425416e84900cc
/tqdm-4.64.1-py2.py3-none-any.whl (78kB)
    100% |████████████████████████████████| 81kB 955kB/s
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from pytorch-pretrained-bert) (2.
18.4)
Collecting boto3 (from pytorch-pretrained-bert)
  Downloading https://files.pythonhosted.org/packages/75/ca/d917b244919f1ebf96f7bbd5a00e4641f7e9191b0d070258f5dc10f5eaad
/boto3-1.23.10-py3-none-any.whl (132kB)
    100% |████████████████████████████████| 133kB 13.0MB/s
Collecting torch>=0.4.1 (from pytorch-pretrained-bert)
  Downloading https://files.pythonhosted.org/packages/c4/49/9da10fef2c2ba8ff91eeab70a123ca60d082b1012b3aff7825c9b1115852
/torch-1.10.2-cp36-cp36m-win_amd64.whl (226.6MB)
```

4.3 界面分析与使用说明

一个独立的数据集包括以下文件

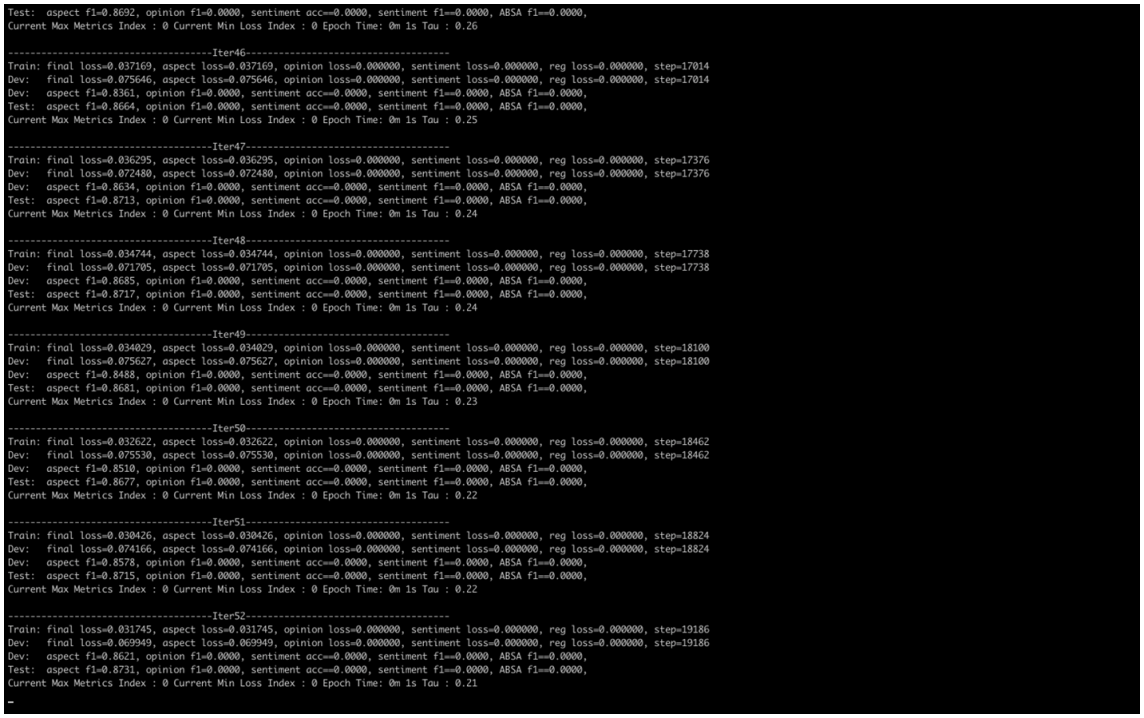
target.txt	2022/12/7 21:44	文本文档	92 KB
sentence.txt	2022/12/7 21:44	文本文档	225 KB
internal_forward_top10.txt	2022/12/7 21:44	文本文档	5,077 KB
internal_backward_top10.txt	2022/12/7 21:44	文本文档	5,185 KB
external_forward_top10.txt	2022/12/7 21:44	文本文档	5,402 KB
external_backward_top10.txt	2022/12/7 21:44	文本文档	5,389 KB
bert_pt_top10.txt	2022/12/7 21:44	文本文档	5,632 KB
bert_base_top10.txt	2022/12/7 21:44	文本文档	5,512 KB

```
(myGNNs) ➜  SoftProto git:(master) ✗ CUDA_VISIBLE_DEVICES=0 python train_softproto.py --dataset res14 --lm external --seed 123
```

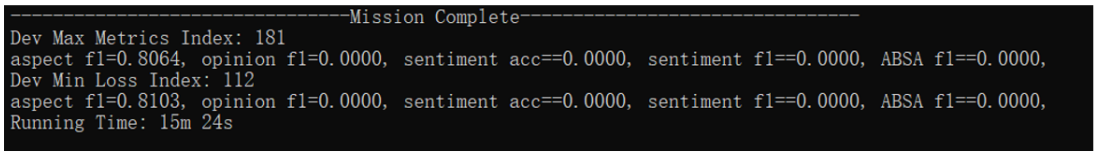
–lm 用于指定预训练语言模块，即对指定的数据集进行 SoftProto 的训练和评估

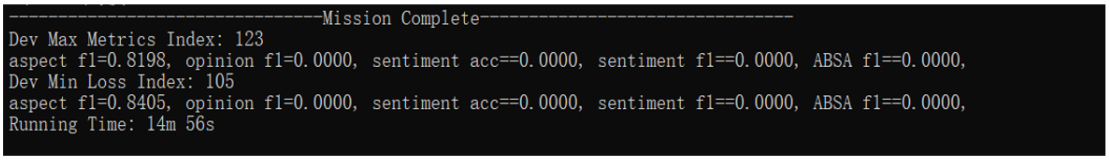
4.4 算法运行时截图

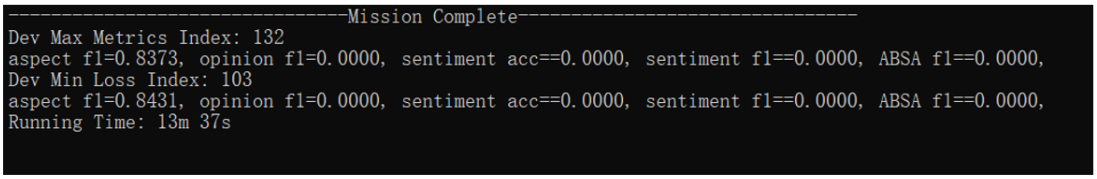
```
CUDA_VISIBLE_DEVICES=0 python train_softproto.py --dataset res14 --lm externa
(myGNNs) ➜  SoftProto git:(master) ✗ CUDA_VISIBLE_DEVICES=0 python train_softproto.py --dataset res14 --lm external --seed 123
Model is DECNN
> log file: ./log/res14/external-res14-221207-123451.log
Reuse Word Dictionary & Embedding
cuda memory allocated: 44653056
n_trainable_params: 2444835, n_nontrainable_params: 8718000
> training arguments:
>>> dataset: res14
>>> model_name: DECNN
>>> batch_size: 8
>>> learning_rate: 0.0001
>>> lr_decay: 1e-05
>>> num_epoch: 200
>>> emb_dim: 400
>>> hidden_dim: 400
>>> keep_prob: 0.5
>>> l2_reg: 1e-05
>>> lm: external
>>> class_num: 3
>>> seed: 123
>>> log_step: 5
>>> valset_num: 150
>>> reuse_embedding: 1
>>> optimizer: <class 'torch.optim.adam.Adam'>
>>> initializer: <function uniform_ at 0x7f955721b040>
>>> device: cuda:0
>>> topk: 7
>>> max_sentence_len: 80
>>> model_class: <class 'models.decnn.DECNN'>
>>> dataset_file: {'train': './data/res14/train/', 'test': './data/res14/test/'}
>>> dataset_path: ./data/res14/
>>> inputs_cols: ['sentence', 'mask', 'position', 'keep_prob']

-------------------------------------Iter0-------------------------------------
Train: final loss=0.524122, aspect loss=0.524122, opinion loss=0.000000, sentiment loss=0.000000, reg loss=0.000000, step=362
Dev:   final loss=0.462499, aspect loss=0.462499, opinion loss=0.000000, sentiment loss=0.000000, reg loss=0.000000, step=362
Dev:   aspect f1=0.3986, opinion f1=0.0000, sentiment acc==0.0000, sentiment f1==0.0000, ABSA f1==0.0000,
Test:  aspect f1=0.4171, opinion f1=0.0000, sentiment acc==0.0000, sentiment f1==0.0000, ABSA f1==0.0000,
Current Max Metrics Index : 0 Current Min Loss Index : 0 Epoch Time: 0m 2s Tau : 1.00

-------------------------------------Iter1-------------------------------------
Train: final loss=0.302238, aspect loss=0.302238, opinion loss=0.000000, sentiment loss=0.000000, reg loss=0.000000, step=724
Dev:   final loss=0.294583, aspect loss=0.294583, opinion loss=0.000000, sentiment loss=0.000000, reg loss=0.000000, step=724
Dev:   aspect f1=0.6026, opinion f1=0.0000, sentiment acc==0.0000, sentiment f1==0.0000, ABSA f1==0.0000,
Test:  aspect f1=0.6103, opinion f1=0.0000, sentiment acc==0.0000, sentiment f1==0.0000, ABSA f1==0.0000,
```



①使用了DECNN 模型作为 SoftProto 框架中的序列标注器，然后分别使用② SoftProtol 和③ SoftProtoE 对其进行增强。







5 实验结果分析

实验结果如图 5所示：

	LAP14	RES14	RES15	RES16
DECNN,round1	80.64	85.77	70.12	75.46
DECNN,round2	79.51	85.01	69.48	74.25
DECNN,round3	80.44	85.66	71.08	74.92
DECNN,round4	79.40	85.23	70.46	75.52
DECNN,round5	79.46	86.12	70.69	74.52
平均值（DECNN）	79.89	85.56	70.37	74.93
+SoftProtoI，round1	81.12	85.99	71.20	75.23
+SoftProtoI，round2	81.25	86.02	70.79	75.36
+SoftProtoI，round3	82.98	86.15	71.62	74.90
+SoftProtoI，round4	82.41	85.46	70.98	75.99
+SoftProtoI，round5	81.38	86.66	71.35	76.05
平均值（+SoftProtoI）	81.83（+1.94）	86.01（+0.55）	71.12（+0.75）	75.56（+0.57）
+SoftProtoE,round1	81.51	86.51	71.97	76.36
+SoftProtoE,round2	82.23	86.32	70.99	75.55
+SoftProtoE,round3	82.90	85.99	72.08	75.92
+SoftProtoE,round4	81.98	86.14	71.46	76.52
+SoftProtoE,round5	82.05	86.12	71.69	75.52
平均值（+SoftProtoE）	82.13(+2.24)	86.22(+0.66)	71.64(+1.27)	75.97(+1.04)

图 5: 实验结果

SoftProto 带来的提升在小数据集（Res15 和 Res16）上更为明显，这是因为在小数据集中没有足够的样本来训练一个好的神经网络序列标注器。同时，SoftProtoE 的性能优于 SoftProtoI，这是因为外部语料库要比 ATE 自身的数据集大得多，在其上训练的语言模型也包含了更多的信息，可以生成质量更高的软模板。

6 总结与展望

本文提出了一种通用的 SoftProto 框架来增强 ATE 任务。相较于设计复杂的序列标注器，我们转向将样本通过软模板相互关联。借助语言模型来自动生成软模板，并设计了一个简单而有效的门控调制器来利用软模板。在 SemEval 四个数据集上的实验表明，SoftProto 显著地提升了三种经典 ATE 模型的性能，并同时维持了较低的计算开销。由于水平有限，只选取了 DECNN 作为模型来判定其性能，而作者选取了 BiLSTM、DECNN 和 Seq2Seq4ATE 三种模型作为 SoftProto 框架中的序列标注器，然后分别使用 SoftProtoI 和 SoftProtoE 对其进行增强。且发现引入如 Yelp 和 Amazon 的外部大规模语料后，SoftProto 的性能还可以进一步提升。

参考文献

[1] CHEN Z, QIAN T. Enhancing Aspect Term Extraction with Soft Prototypes[J]., 2020, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP): 2107-2117.