Implementation Tools for Information Extraction

Bruno Seo (sbseo2)

Introduction

In this technology review, we examine a variety of tools used for information extraction. We will primarily focus on Named Entity Recognition (NER) tools, one of the subtasks of sequence labeling. While there are many tools available online, this review suggests using two tools. One is Spacy, and the other is LM-LSTM-NER. We will compare their pros and cons and future directions.

Body

1. Named Entity Recognition

NER is the task of extracting entities when given sentences. For example, when given a sentence such as "My name is Bruno Seo," its task is to extract entities such as noun and verb. Here, nouns are `name` and `Bruno Seo`, and a verb is `is`. Therefore, the output of NER will be `noun: name, Bruno Seo` and `verb: is`. Note that a user can freely set up entities on their own. Entities can not only be `part-of-speech tagging` but also `subject`, `research area`, `location`, `email address`, and so on.

2. Spacy

One reason to use Spacy is because of its simplicity. Its installation process is short and concise. The following bash command can easily install it.

```
pip install spacy
```

To perform NER, the only step left now is to construct a training dataset. Dataset can be constructed by the following manner. Note that train dataset consists of a sentence, entity type, and its position in a given string.

```
TRAIN_DATA = [
    ('My main research interests are in machine learning, artificial
intelligence, and theoretical computer science.', {
        'entities': [(34,50, 'AREA'), (52, 75, 'AREA'), (81, 109, 'AREA')]
    }),
    ('My primary research areas are computational Biology, Bioinformatics and
Machine learning.', {
        'entities': [(53,67, 'AREA'), (72,88, 'AREA')]
    })
]
```

While NER in Spacy is easy to use, it supports only pre-trained CNN language models. Therefore, NER in Spacy is ideal for building a prototype quickly. However, Spacy is not an ideal tool for competitive tasks when compared to a state-of-arts language model.

3. LM-LSTM-CRF

LM-LSTM-CRF is one of the state-of-the-art models for the information extraction task. For a task that requires a powerful language model, this model is the way to go. This model leverages both the character-level language model and the word-level language model. LSTM enables recurrence so that the model can capture the contexts in a text. Typically, LSTM is known to outperform CNN when it comes to capturing context.

Since this tool is not available in pip, we need to clone it from its website directly. This can be done by the following bash command.

```
git clone https://github.com/LiyuanLucasLiu/LM-LSTM-CRF
```

To train this dataset, we need to label the entities as the same as the CoNLL 2003 NER dataset format. Note that `empty lines` are used as separators between sentences. Additionally, `-DOCSTART- -X- -X- -X- O` is used separators between documents.

```
-DOCSTART- -X- -X- -X- O

Pierre NNP
Vinken NNP
, ,
61 CD
```

While LM-LSTM-CRF provides a powerful outcome, it also requires an expensive computational cost. According to the authors, training a model for CoNLL03 NER completes in about 6 hours on a single GPU.

Conclusion

To build an easy and quick prototype, Spacy is an ideal tool for NER. It is easy to install and requires low computational cost. It also provides a small-sized pre-trained CNN language model.

To obtain a competitive outcome, LM-LSTM-CRF is an ideal tool. It takes more time for installation and requires a higher computational cost. It also provides pre-train LSTM language models. It also supports training a custom language model, but it is computationally expensive.

Reference

Spacy

- https://towardsdatascience.com/train-ner-with-custom-training-data-using-spacy-525ce748fab7

LM-LSTM-CRF

- https://github.com/LiyuanLucasLiu/LM-LSTM-CRF
- https://arxiv.org/abs/1709.04109