SummerTime: A Text Summarization Platform

Ansong Ni, **Zhangir Azerbayev**, Murori Mutuma, Troy Feng, Yusen Zhang, Dragomir Radev

Yale

LILY Lab

Introduction

Recent years have seen strong progress in document summarization. Large transformer-based models such as PEGASUS have improved abstractive summarization, and more attention has been paid to high quality data in domains such as multi-document, conversational, and long-form document summarization. However, this presents engineers seeking to deploy summarization systems with a challenge. How do you choose between the variety of summarization models and make complex tradeoffs between efficiency and performance? What models are best suited for my dataset? How do I interpret the array of summarization metrics that each have their flaws? Furthermore, popular NLP tools such as AllenNLP and Huggingface are unfriendly to engineers without domain expertise in NLP. The SummerTime library seeks to solve these problems by creating a user-friendly platform for models, data, and evaluation for document summarization.

Models

Model selection in SummerTime works under the principle of progressive disclosure of complexity. When the user does not choose to leverage specific knowledge about NLP or their domain, that complexity is hidden from them. At the highest level, users can load a default model by calling `st.model.summarizer`, which chooses a reasonable default summarization model, and interact with the high-level API consisting of methods such as `model.summarize()`. If they wish, users can also load domain-specific models such as `st.model.multi_document_summarizer()`. Users with NLP and machine learning knowledge can select specific models, for example `st.model.pegasus()` and work with the underlying pytorch module.

Integrated into the models are also helpful documentation features that help users evaluate trade-offs between different summarization models.

Figure 1: SummerTime module structure

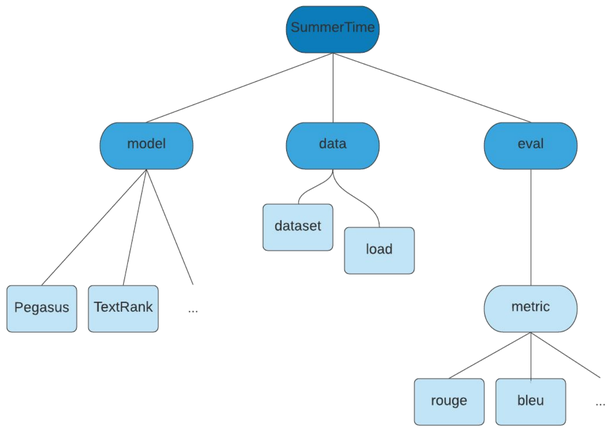

Figure 2: Code examples

```
# Users can load a default summarization model
model = st_model.summarizer()

# Or a specific model
pegasus = st_model.pegasus()
```

```
# Initializes a rouge metric object
metric = st_eval.rouge()

# Evaluates model on subset of cnn_dailymail
# Stores various versions of rouge as an object variable
metric.evaluate(model, cnn_dataset['test'][0:5]);
```

```
# Retrieve rouge f-scores.
metric.get(['rouge_1_f_score', 'rouge_2_f_score', 'rouge_l_f_score'])
```

```
# Users can easily access documentation to assist with model selection
model.show_capability()
```

```
Pegasus is the default singe-document summarization model.
Introduced in 2019, a large neural abstractive summarization model trained on web crawl and news data.
 Strengths:
 - High accuracy
 - Performs well on almost all kinds of non-literary written text
 Weaknesses:
 - High memory usage
 Initialization arguments:
 - `device = 'cpu'` specifies the device the model is stored on and uses for computation. Use `device='gpu'` to run on an Nv
idia GPU.
```

Data

SummerTime provides a common dataset API that makes loading, storing, and evaluating on data fast and user-friendly. Users can either download common summarization datasets such as CNN-Dailymail, or load their own data into a dataset object. The common API makes evaluating the same model on different datasets or evaluating a suite of models on a dataset fast and simple.

Evaluation

Evaluation is perhaps the most difficult step in the document summarization pipeline. There is a wide variety of summarization metrics, for each of which a high-score is not guaranteed to correspond to high quality summaries. SummerTime's evaluation module uses SummEval as a backend, and is an effort to make that library accessible to engineers without an NLP background.

Advanced Features

Beyond basic model, data, and evaluation features, the SummerTime team seeks to implement several more technically-challenging features in the future.

One feature we wish to implement is post-training length control. Supervised summarization models implicitly learn desired summary lengths from the training set. However, in real-world model deployment there is no guarantee that the summary length learned at training-time matches the desired summary length for an application. Additionally, users may want to create different-length summaries of the same document for various applications. The SummerTime team is exploring various techniques for post-training summary length control.

We are also exploring techniques for automatic error analysis. In applications, the kinds of errors a model makes may be as important as how often it makes them. For example, a summarizer hallucinating false information that is not in the source document might be much worse than other errors such as repeating information.