SOTASTREAM: A Streaming Approach to Machine Translation Training

Matt Post, Thamme Gowda, Roman Grundkiewicz,
Huda Khayrallah, Rohit Jain, Marcin Junczys-Dowmunt

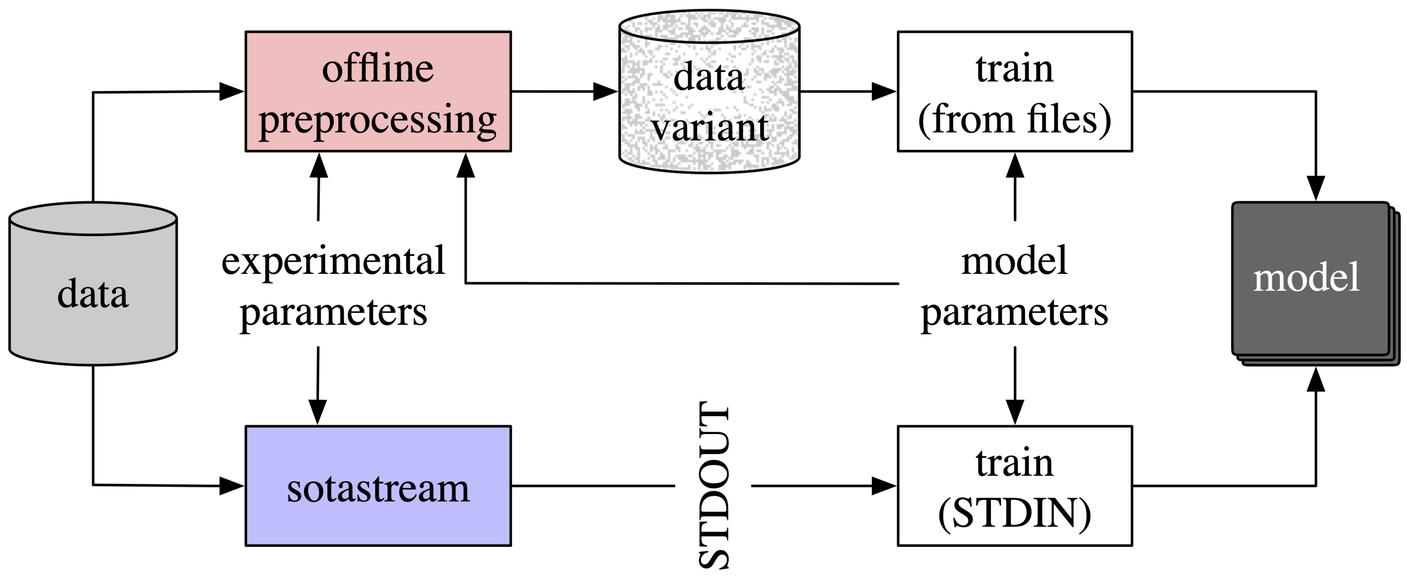
Introduction

Standard off-line data preparation is expensive...

Data tensorized ahead of time:
- takes up time and disk space,
- ties the prepared dataset to a model configuration (e.g., vocabulary).

The solution: generate data dynamically!

On-the-fly data generation:
- decouples data preparation from model training,
- enables the standard UNIX command API as UI.



SOTASTREAM

Samples data from sharded pools and generates an infinite stream of permutations over those pools. Streams can be easily mixed and perturbed with chains of composable augmentations.

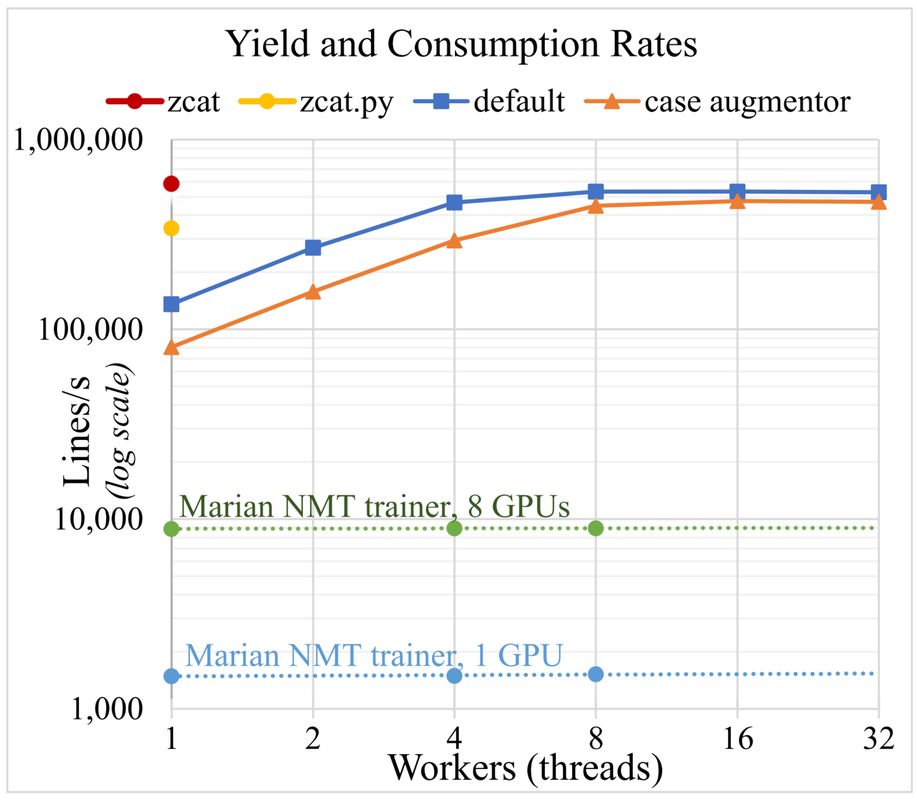
Use cases

Mixing multiple streams of data	Data augmentation for robustness	Filtering bad data examples
Subword tokenization sampling	Training document-context models	Alignments and other data types
Data collection tools: e.g., mtdata	Generating datasets for offline use	...

Usage example

```python
@pipeline("robust-case")
class RobustCasePipeline(Pipeline):
  def __init__(self, pa_dir: str, bt_dir: str, **kwargs):
    super().__init__(**kwargs)
    pa_stream = self.create_data_stream(pa_dir, processor=Augment)
    bt_stream = self.create_data_stream(bt_dir,
      processor=partial(Augment, tag="[BT]"))  # tag the BT data
    self.stream = Mixer([pa_stream, bt_stream], self.mix_weights)
  # definitions of other class methods go here ...

def LowerCase(stream: Generator[Line]) -> Generator[Line]:
  for line in stream:
    line[0] = line[0].lower()  # lowercase the source side
    yield line

def TitleCase(stream: Generator[Line]) -> Generator[Line]:
  for line in stream:
    line[0], line[1] = line[0].title(), line[1].title()  # titlecase both sides
    yield line

def TagData(stream: Generator[Line], tag: str) -> Generator[Line]}:
  for line in stream:
    line[0] = f"{tag} {line}"  # add a target language tag to the source
    yield line

def Augment(path: str, tag: str = None) -> Generator[Line]:
  stream = UTF8File(path)   # open the path to the shard

  stream = Mixer(  # randomly mix casing variants
    [ stream, LowerCase(stream), TitleCase(stream) ],
    [ 0.95,   0.04,              0.01 ],
  )

  if tag is not None:
    stream = TagData(stream, tag)
  return stream
```

Benchmarks



Model	English–German (newstest2021) COMET20	BLEU	chrF	Czech–Ukranian (wmttest2022) COMET22	BLEU	chrF
Best constrained	54.8	31.3	60.7	91.6	34.7	61.5
Full loading	55.9 ± 0.4	34.9 ± 0.1	62.0 ± 0.0	85.5 ± 0.2	27.9 ± 0.4	55.6 ± 0.2
Sequential streaming	56.1 ± 0.2	35.0 ± 0.2	62.1 ± 0.0	86.4 ± 0.1	28.7 ± 0.3	56.6 ± 0.2
Randomized streaming	55.8 ± 0.2	35.1 ± 0.0	62.2 ± 0.0	85.6 ± 0.1	27.8 ± 0.0	55.6 ± 0.2
SOTASTREAM	55.9 ± 0.1	34.9 ± 0.1	62.1 ± 0.1	85.7 ± 0.2	28.5 ± 0.4	56.2 ± 0.2

Model	Time (Hours)
Full loading	36.84 ± 0.16
Sequential streaming	35.51 ± 0.15
Randomized streaming	35.73 ± 0.05
SOTASTREAM	35.86 ± 0.27

The dynamic data generation in SOTASTREAM:

- ✓ Just as accurate
- ✓ Just as fast
- ✓ Saves disk space
- ✓ More flexible

`pip install sotastream` | https://github.com/marian-nmt/sotastream | MIT License