

Overlapping with language modelling

Code to reproduce the results from "Alleviating Sequence Information Loss with Data Overlapping and PrimeBatch Sizes".

Hold experiments on 4 models using the overlapping:

- **awd**, [AWD](#) ASGD Weight-Dropped LSTM, ([/awd](#))
- **text simple**, a very basic lstm for language modelling, ([/simple](#))
- **mos**, [MOS](#) Mixture of Softmaxes, ([/mos](#))
- **voice simple**, a very basic LSTM for emotion detection on voice, ([/emotions](#))

To specify which model to run, use `--main-model {simple-lstm | awd-lstm | mos-lstm | emotions-simple-lstm}` argument. There are additional common parameters, as well as specific parameters for each model. Those can be found in [main_run.py](#).

The taxonomy in the code may differ a bit from the paper, especially regarding the type of experiments. Here is the corresponding terms:

In the code	In the paper
No order	Extreme TOI
Local order	Inter-batch TOI
Standard order	Standard TOI
Total order (P)	Alleviated TOI (P)

Set-up

Download the data (PTB, WT2, WT103):

```
chmod +x get_data.sh
./get_data.sh
```

For emotions, add in [data/IEMOCAP/](#) the [all_features_cv](#) files.

We use python [3.6](#) with Pytorch [0.4.1](#). To create a new python environment and install dependencies, run:

```
python3.6 -m virtualenv venv
source venv/bin/activate
pip3 install -r requirements.txt
```

About the files

`main_run.py` is the main entry point that parses arguments, does the global initialization and runs the corresponding model and task.

`awd/`, `emotions/`, `mos/` and `simple/` are the different models directories. `common/` holds the common initialization and utilities, such as the different data iterators, which are in the `DataSelector` class in `common/excavator.py`.

The `main_run.py` file, after performing the common initializations, imports the `main.py` file corresponding to the chosen model.

Commands to reproduce the experiments

Note: Those results do not use prime batch size, but the default parameters. To have better results, adapt the `--batch-size` param to the closest prime number.

Quick anchors navigation:

Model	Dataset	Experiments
AWD	PTB	Extreme / Inter-batch / Original / Alleviated TOI
	WT2	Extreme / Inter-batch / Original / Alleviated TOI
	WT103	Extreme / Inter-batch / Original / Alleviated TOI
Text simple LSTM	PTB	Extreme / Inter-batch / Original / Alleviated TOI
	WT2	Extreme / Inter-batch / Original / Alleviated TOI
MOS	PTB	Original / Alleviated TOI
Voice simple LSTM	IEMOCAP	Extreme / Inter-batch / Original / Alleviated TOI

AWD PTB

Extreme TOI:

Expected results: **66.38** / **63.49** (validation / testing)

```
python3 main_run.py --main-model awd-lstm --batch-size 20 --data data/penn
--dropouti 0.4 --dropouth 0.25 --seed 141 --seed-shuffle 141 --epochs 1000
--shuffle-full-seq
```

Inter-batch TOI:

Expected results: **66.96** / **64.20** (validation / testing)

```
python3 main_run.py --main-model awd-lstm --batch-size 20 --data data/penn
--dropouti 0.4 --dropouth 0.25 --seed 141 --seed-shuffle 141 --epochs 1000
--shuffle-row-seq
```

Standard TOI:

Expected results: **61.28** / **58.94** (validation / testing)

```
python3 main_run.py --main-model awd-lstm --batch-size 20 --data data/penn
--dropouti 0.4 --dropouth 0.25 --seed 141 --epochs 1000
```

Alleviated TOI {2,5,7,10}:

Expected results (validation / testing):

- 2: **61.73** / **59.37**
- 5: **63.37** / **60.50**
- 7: **59.22** / **56.7**
- 10: **68.09** / **65.88**

```
overlaps=(2 5 7 10)
epochs=1000
for k in "${overlaps[@]}"
do
    :
    python3 main_run.py --main-model awd-lstm --batch-size 20 --data
data/penn --dropouti 0.4 --dropouth 0.25 --seed 141 --epochs
"$((($epochs/$k))" --init-seq "overlapCN_{$k}"
    sleep 10
done
```

AWD WT2**Extreme TOI**

Expected results: **77.14** / **73.52** (validation / testing)

```
python3 main_run.py --main-model awd-lstm --epochs 750 --data
/data/noemien.kocher/datasets/wikitext-2 --dropouth 0.2 --seed 1882 --
batch-size 80 --shuffle-full-seq
```

Inter-batch TOI

Expected results: **76.08** / **72.61** (validation / testing)

```
python main_run.py --main-model awd-lstm --epochs 750 --data
/data/noemien.kocher/datasets/wikitext-2 --dropouth 0.2 --seed 1882 --
batch-size 80 --shuffle-row-seq
```

Standard TOI

Expected results: **68.50** / **65.86** (validation / testing)

```
python3 main_run.py --main-model awd-lstm --epochs 750 --data
/data/noemien.kocher/datasets/wikitext-2 --dropouth 0.2 --seed 1882 --
batch-size 80
```

Alleviated TOI {2,5,7,10}

Expected results (validation / testing):

- 2: **68.56** / **65.51**
- 5: **69.56** / **66.33**
- 7: **67.48** / **64.87**
- 10: **72.95** / **69.69**

```
overlaps=(2 5 7 10)
epochs=750
for k in "${overlaps[@]}"
do
    :
    python3 main_run.py --main-model awd-lstm --data
/data/noemien.kocher/datasets/wikitext-2 --dropouth 0.2 --seed 1882 --
batch-size 80 --epochs "$(($epochs/$k))" --init-seq "overlapCN_${k}"
    sleep 10
done
```

AWD WT103

Extreme TOI

Expected results: **35.22** / **36.19** (validation / testing)

```
python3 -u main_run.py --main-model awd-lstm --epochs 14 --nlayers 4 --
emsize 400 --nhid 2500 --alpha 0 --beta 0 --dropoute 0 --dropouth 0.1 --
dropouti 0.1 --dropout 0.1 --wdrop 0 --wdecay 0 --bptt 140 --batch-size 60
--optimizer adam --lr 1e-3 --data /data/noemien.kocher/datasets/wikitext-
103 --when 12 --model QRNN --shuffle-full-seq
```

Inter-batch TOI

Expected results: **35.41** / **36.39** (validation / testing)

```
python3 -u main_run.py --main-model awd-lstm --epochs 14 --nlayers 4 --
emsize 400 --nhid 2500 --alpha 0 --beta 0 --dropoute 0 --dropouth 0.1 --
dropouti 0.1 --dropout 0.1 --wdrop 0 --wdecay 0 --bptt 140 --batch-size 60
--optimizer adam --lr 1e-3 --data /data/noemien.kocher/datasets/wikitext-
103 --when 12 --model QRNN --shuffle-row-seq
```

Standard TOI

Expected results: **32.18** / **32.94** (validation / testing)

```
python3 -u main_run.py --main-model awd-lstm --epochs 14 --nlayers 4 --
emsize 400 --nhid 2500 --alpha 0 --beta 0 --dropoute 0 --dropouth 0.1 --
dropouti 0.1 --dropout 0.1 --wdrop 0 --wdecay 0 --bptt 140 --batch-size 60
--optimizer adam --lr 1e-3 --data /data/noemien.kocher/datasets/wikitext-
103 --when 12 --model QRNN
```

Alleviated TOI {2,5,7,10}

Expected results (validation / testing):

- 2: **36.94** / **34.31**
- 5: **38.50** / **40.04**
- 7: **31.78** / **32.72**
- 10: **48.28** / **49.49**

```
# base num epochs is 14
overlaps=(2 5 7 10)
when_steps=147456
max_steps=172032
for i in "${!overlaps[@]}"
do
    :
    python3 -u main_run.py --main-model awd-lstm --epochs 14 --nlayers
4 --emsize 400 --nhid 2500 --alpha 0 --beta 0 --dropoute 0 --dropouth 0.1
--dropouti 0.1 --dropout 0.1 --wdrop 0 --wdecay 0 --bptt 140 --batch-size
60 --optimizer adam --lr 1e-3 --data
/data/noemien.kocher/datasets/wikitext-103 --when-steps "$when_steps" --
model QRNN --init-seq "overlapCN_${overlaps[$i]}" --log-dir
/data/noemien.kocher/logs/ --max-steps "$max_steps"
    sleep 10
done
```

Simple PTB

Extreme TOI:

Expected results: **81.97** / **79.08** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 20
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1500 --lr-decay 1 --shuffle-
full-seq
```

Inter-batch TOI:

Expected results: **81.67** / **78.59** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 20
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1500 --lr-decay 1 --shuffle-
row-seq
```

Standard TOI:

Expected results: **77.54** / **75.36** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 20
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1500 --lr-decay 1
```

Alleviated TOI {2,5,7,10}:

Expected results (validation / testing):

- 2: **78.48** / **76.55**
- 5: **91.95** / **89.64**
- 7: **77.47** / **74.98**
- 10: **92.92** / **92.07**

```
overlaps=(2 5 7 10)
epochs=100
for k in "${overlaps[@]}"
do
    :
    python3 main_run.py --main-model simple-lstm --epochs
"$((($epochs/$k))" --batch-size 20 --dropout 0.15 --nlayers 2 --bptt 70 --
nhid 1500 --lr-decay 1 --init-seq "overlapCN_{$k}"
    sleep 10
done
```

Simple WT2

Extreme TOI

Expected results: **101.3** / **96.08** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 80
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1150 --lr-decay 1 --data
/data/noemien.kocher/datasets/wikitext-2 --shuffle-full-seq
```

Inter-batch TOI

Expected results: **101.7** / **96.89** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 80
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1150 --lr-decay 1 --data
/data/noemien.kocher/datasets/wikitext-2 --shuffle-row-seq
```

Standard TOI

Expected results: **98.85** / **93.15** (validation / testing)

```
python3 main_run.py --main-model simple-lstm --epochs 100 --batch-size 80
--dropout 0.15 --nlayers 2 --bptt 70 --nhid 1150 --lr-decay 1 --data
/data/noemien.kocher/datasets/wikitext-2
```

Alleviated TOI {2,5,7,10}

Expected results (validation / testing):

- 2: **100.4** / **94.49**
- 5: **113.5** / **106.1**
- 7: **98.25** / **92.77**
- 10: **151.0** / **135.1**

```
overlaps=(2 5 7 10)
epochs=100
for k in "${overlaps[@]}"
do
    :
    python3 main_run.py --main-model simple-lstm --epochs
"$((($epochs/$k))" --batch-size 80 --dropout 0.15 --nlayers 2 --bptt 70 --
nhid 1150 --lr-decay 1 --data /data/noemien.kocher/datasets/wikitext-2 --
init-seq "overlapCN_${k}"
    sleep 10
done
```

MOS PTB

Standard TOI:

Expected results: **58.49** / **56.19** (validation / testing)

```
python3 main_run.py --main-model mos-lstm --data data/penn --dropouti 0.4
--dropoutl 0.29 --dropouth 0.225 --seed 28 --batch-size 12 --lr 20.0 --
epochs 1000 --nhid 960 --nhidlast 620 --emsize 280 --n-experts 15
```

Alleviated TOI {2,5,7,10}:

Expected results (validation / testing):

- 7: **57.34** / **55.09**

```
python3 main_run.py --main-model mos-lstm --data data/penn --dropouti 0.4
--dropoutl 0.29 --dropouth 0.225 --seed 28 --batch-size 12 --lr 20.0 --
epochs 1000 --nhid 960 --nhidlast 620 --emsize 280 --n-experts 15 --init-
seq overlapCN_7
```

Voice simple LSTM

Extreme TOI:

Expected result: **0.475** / **0.377** (WA / UA)

```
python3 main_run.py --main-model emotions-simple-lstm --cv 5 --data
data/IEMOCAP/all_features_cv/ --test-batch-size 1 --lr 0.1 --log-interval
20 --lr-decay 1 --order complete_random
```

Inter-batch TOI:

Expected result: **0.478** / **0.386** (WA / UA)

```
python3 main_run.py --main-model emotions-simple-lstm --cv 5 --data
data/IEMOCAP/all_features_cv/ --test-batch-size 1 --lr 0.1 --log-interval
20 --lr-decay 1 --order local_order
```

Standard TOI:

Expected result: **0.486** / **0.404** (WA / UA)

```
python3 main_run.py --main-model emotions-simple-lstm --cv 5 --data
data/IEMOCAP/all_features_cv/ --test-batch-size 1 --lr 0.1 --log-interval
20 --lr-decay 1 --order standard_order
```


Alleviated TOI 10:

Expected result:

- 15k steps: **0.553** / **0.489** (WA / UA)
- 60 epochs: **0.591** / **0.523** (WA / UA)

```
python3 main_run.py --main-model emotions-simple-lstm --cv 5 --data
data/IEMOCAP/all_features_cv/ --test-batch-size 1 --lr 0.1 --log-interval
20 --lr-decay 1 --order total_order
```

Delayed-reset standard TOI {1,2,5,7,10} with PTB

Expected results (validation / testing):

- 1: **61.28** / **58.94**
- 2: **60.76** / **58.55**
- 5: **60.10** / **57.83**
- 7: **60.08** / **57.76**
- 10: **60.05** / **57.78**

```
P=(1 2 5 7 10)
epochs=1000
for k in "${P[@]}"
do
    :
    python3 main_run.py --main-model awd-lstm-repetitions --batch-size 20
--data data/penn --dropouti 0.4 --dropouth 0.25 --seed 141 --epochs 1000 -
-use-repetitions "${k}"
    sleep 10
done
```

Acknowledgements

Code is heavily borrowed from the following sources:

- simple-lstm (**simple/**): <https://github.com/deeplearningathome/pytorch-language-model>
- awd-lstm (**awd/**): <https://github.com/salesforce/awd-lstm-lm>
- mos-lstm: (**mos/**) <https://github.com/zihangdai/mos>