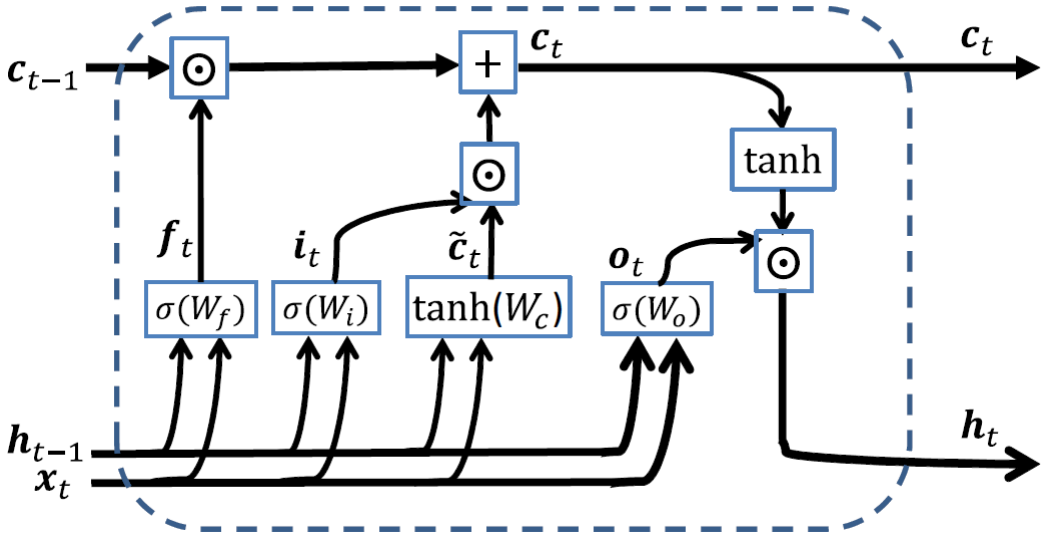
MGU2 Code Exploration

Deliang Yang

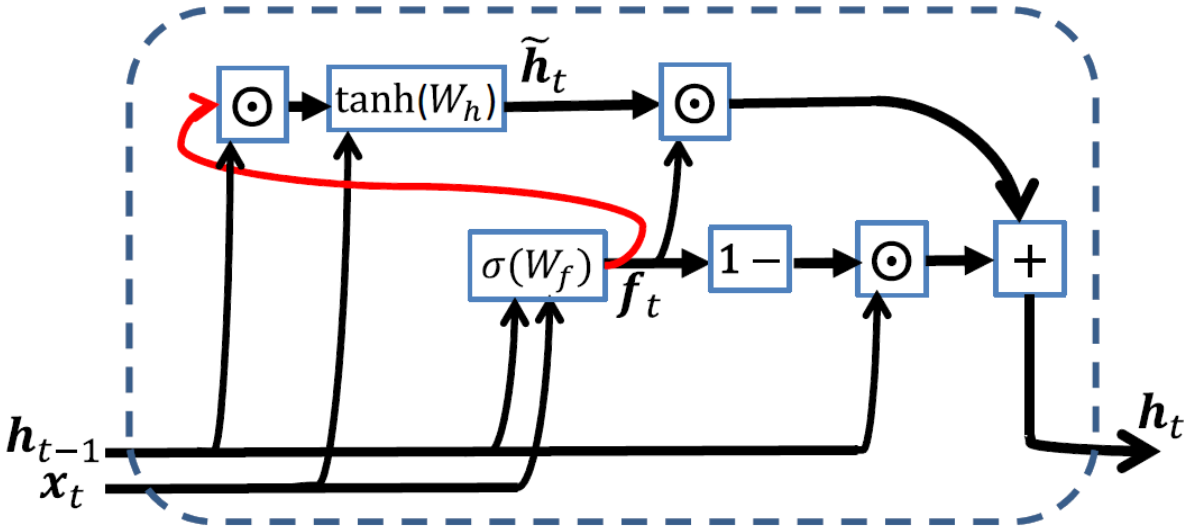
Outline

- Network & Structure
- Code Variation
- Dataset Overview
- Evaluation Result
- Conclusion

Network & Structure – MGU



LSTM



Minimal Gated Unit

[1] Minimal Gated Unit for Recurrent Neural Networks, Zhou et. al. 2016

Network & Structure – Neural Network

consume_less in Keras 1.x

output_dim in Keras 1.x

```
model = Sequential()
if model_name == 'lstm':
    model.add(LSTM(implementation=1, units=100,
                   activation='tanh', input_shape=(64, 64)))
elif model_name == 'basic':
    mgu_basic = MGUBasicModel(implementation=1, units=100,
                    activation='tanh', input_shape=(64, 64))
    model.add(mgu_basic)
elif model_name == 'variant':
    mgu_variant = MGUVariantModel(implementation=1, units=100,
                activation='tanh', input_shape=(64, 64))
    model.add(mgu_variant)

model.add(Dense(CLS_NUM))
model.add(Activation('softmax'))
my_optimizer = RMSprop(lr=0.002)
```

Code Variation

LSTM

```
x_i
x_f
x_c
x_o
i
f
c
o
h
```

Standard MGU

```
x_i
x_c
i
c
h
```

MGU without input and bias

```
x_c
i
c
h
```

$$x_i = W_i x + b_i$$
$$Gate_i = \sigma(U_i h_{t-1} + x_i)$$

Dataset Overview

- MNIST
  - Handwritten digits (0-9)
  - 60,000 training, 10,000 test samples
- IMDB
  - Movie review (sequence): positive or negative
  - Binary classification
  - 25000 training, 25000 test sequences
- NIST
  - Superset of MNIST, handwritten characters (‘A-Z’, ‘a-z’, ‘0-9’, 62 classes)
  - Originally 800k pictures
  - Use a subset to save time (63488 train, 15872 test)

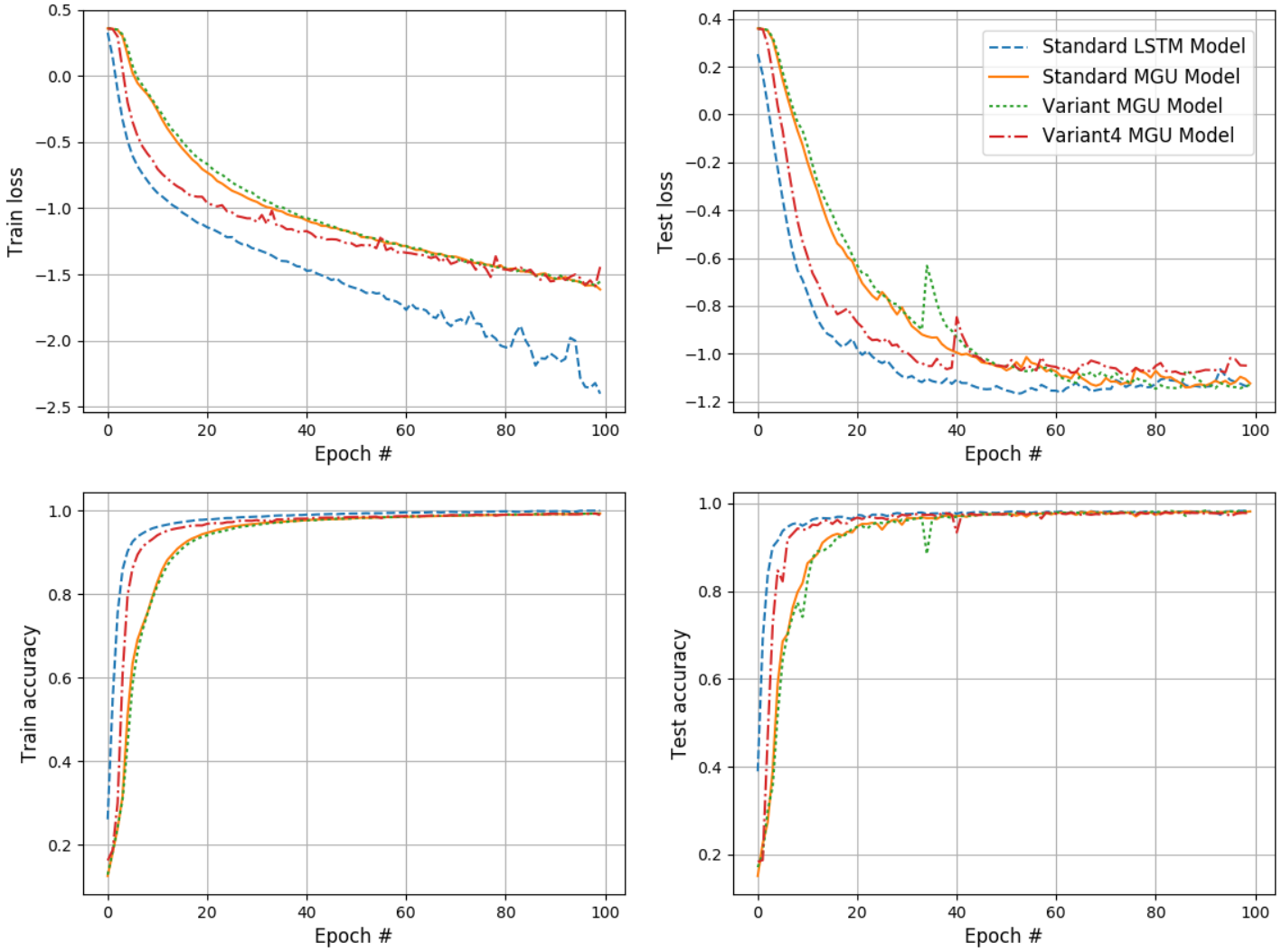
1024 train, 256 test per class

Evaluation Setting

- Batch size: 128
- Epochs: 100
- RNN layer activation: tanh
- Hidden unit size: 100
- LR, optimizers are dataset-dependent

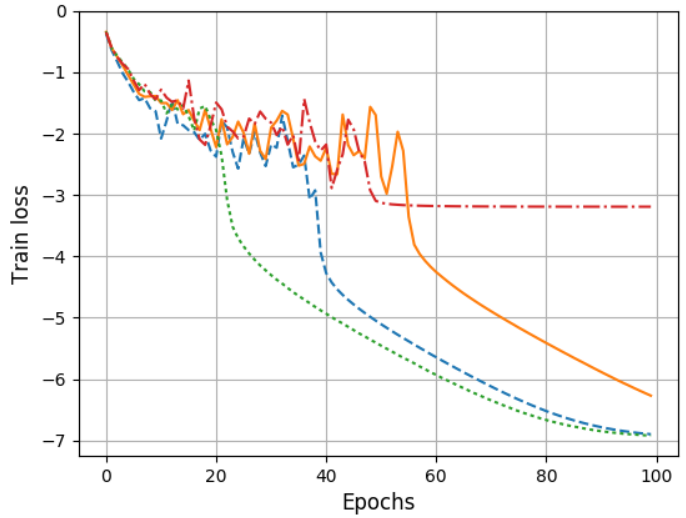
Evaluation on MNIST Dataset

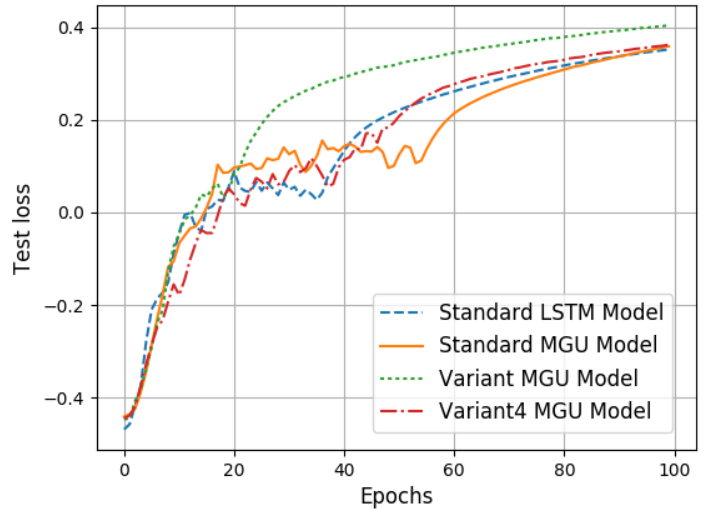
- Settings
  - Activation: Softmax
  - Optimizer: SGD
  - LR = 0.03
- Training converge rate:
  - LSTM > V4 > Std MGU > V
- All most the same test accuracy

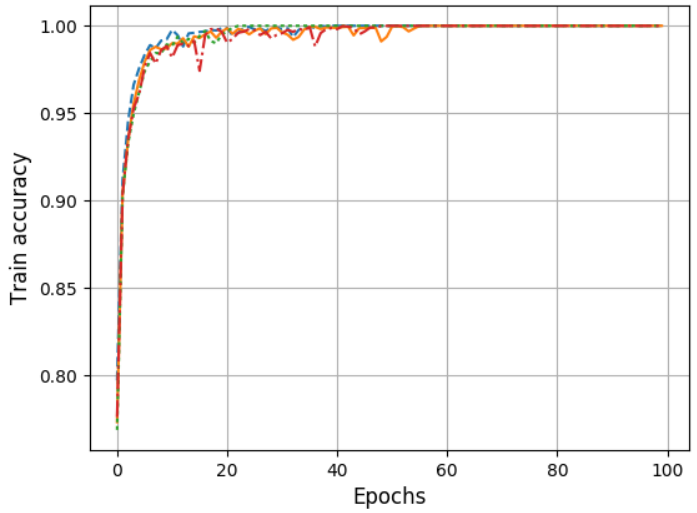


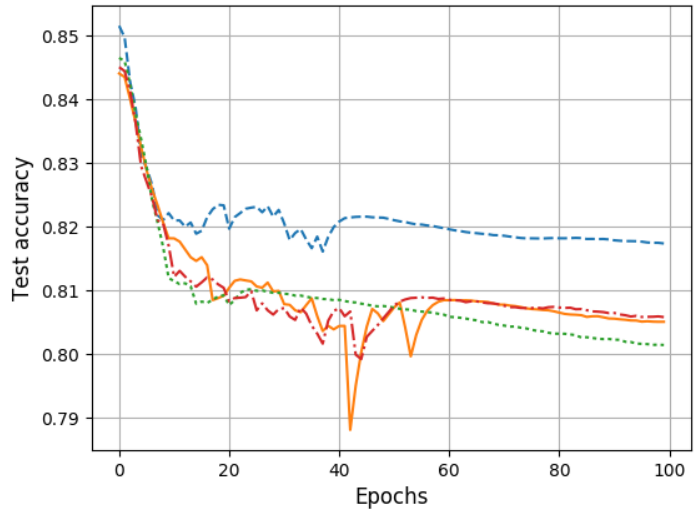
Evaluation on IMDB

- Settings
  - Activation: Sigmoid
  - Optimizer: Adam
  - Learning rate: 0.001
- Training converge rate:
  - V4 > LSTM > Std MGU
- Test acc:
  - LSTM > Std MGU > V
- Std MGU has more fluctuations



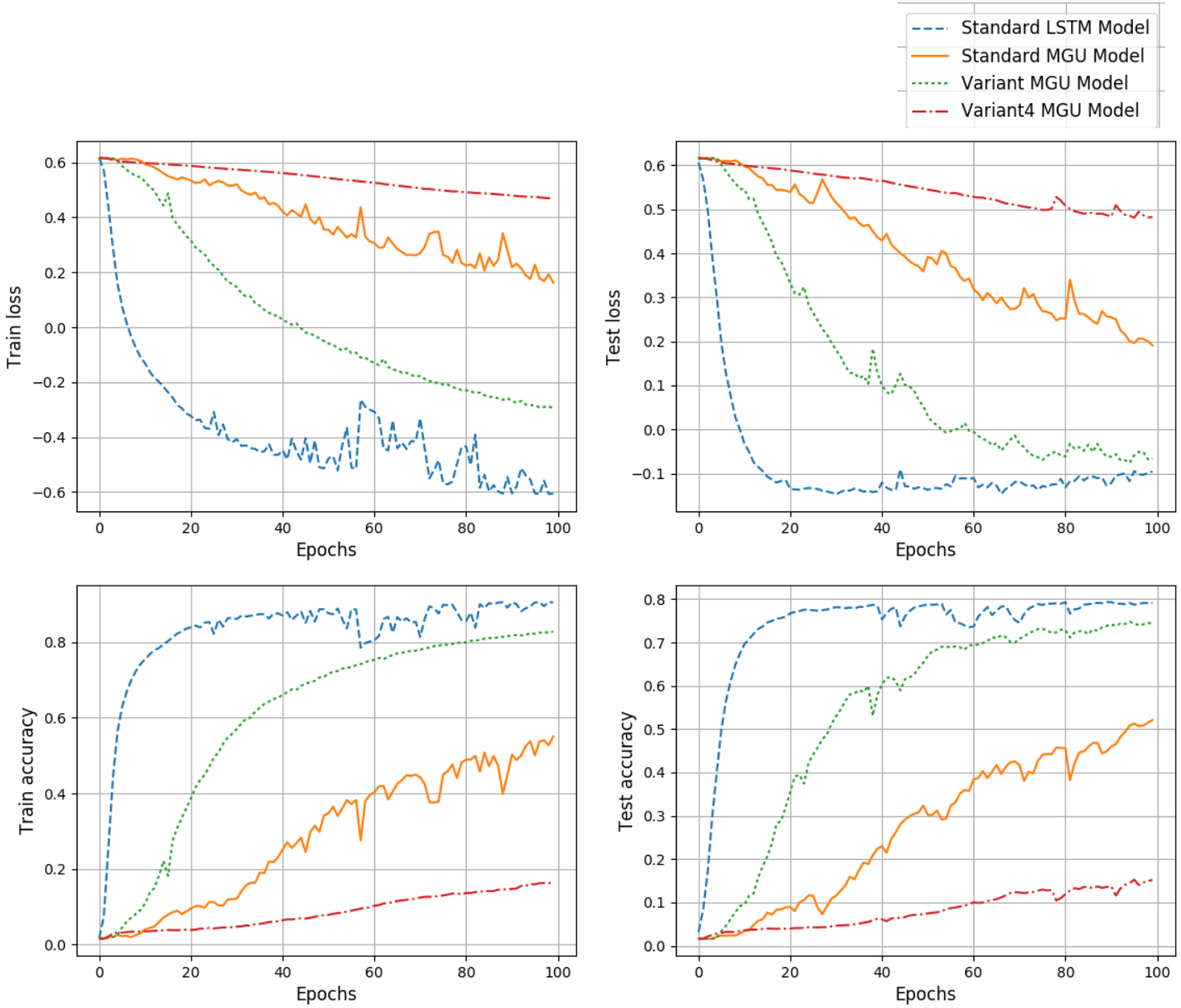






Evaluation on NIST

- Settings
  - Activation: Softmax
  - Optimizer: RMSprop
  - Learning rate = 0.002
- Hard to achieve high acc:
  - High variance within same class
  - `O-o-0, L-l-1-I, P-p, C-c, J-j, K-k, S-s, U-u, V-v …`
- MGU Variant 4 doesn't converge with RMSprop
- V4 uses Adadelta



Training Time Comparison

Dataset	Model	Training Time (s) (100 epochs)	Train Loss	Train Accuracy (%)	Test Loss	Test Accuracy (%)
MNIST	Std LSTM	1618.9	0.0039	99.96	0.0716	98.36
	Std MGU	965.8	0.0242	99.27	0.0700	98.18
	MGU2	904.0	0.0279	99.20	0.0759	98.14
	MGU4	1002.6	0.0360	98.90	0.0891	97.83
IMDB	Std LSTM	3991.1	1.266 e-7	100	2.2649	81.72
	Std MGU	2411.8	5.385 e-7	100	2.3033	80.51
	MGU2	2040.7	1.207 e-7	100	2.5497	80.14
	MGU4	1835.3	6.450 e-4	100	2.3226	80.57
NIST	Std LSTM	3991.7	0.2477	90.50	0.8127	79.01
	Std MGU	2466.2	1.4538	**55.08**	**1.5014**	**53.13**
	MGU2	2195.0	0.5062	82.78	0.8653	74.38
	MGU4 *	2458.5	2.9426	**16.58**	**3.0412**	**15.45**

Training time ratio: 1.85 : 1.12 : 1.0 : ?

(Data come from the final epoch)

Summary

- Removing gates may result in worse performance, LSTM performs best on every dataset.
- Removing gates consumes less time under the same conditions for different models.
- Simple model would suffer from more fluctuations (not robust)
- MGU2 is similar to MGU standard model, but consumes less time
- MGU has trouble with large classes classification problems (model used in the experiment is too simple)

Thanks!