Научно-технологический университет Сириус

СОВРЕМЕННЫЕ МЕТОДЫ ТЕОРИИ ИНФОРМАЦИИ, ОПТИМИЗАЦИИ И УПРАВЛЕНИЯ

Построение помехоустойчивых кодов с помощью машинного обучения

- **Определения**
- Введение
- Проблематика
- Решение
- Результаты
- Выводы

Определения

- AE
- RNN
- AWGN
- BER/BLER

- Определения
- Введение
- Проблематика
- Решение
- Результаты
- Выводы

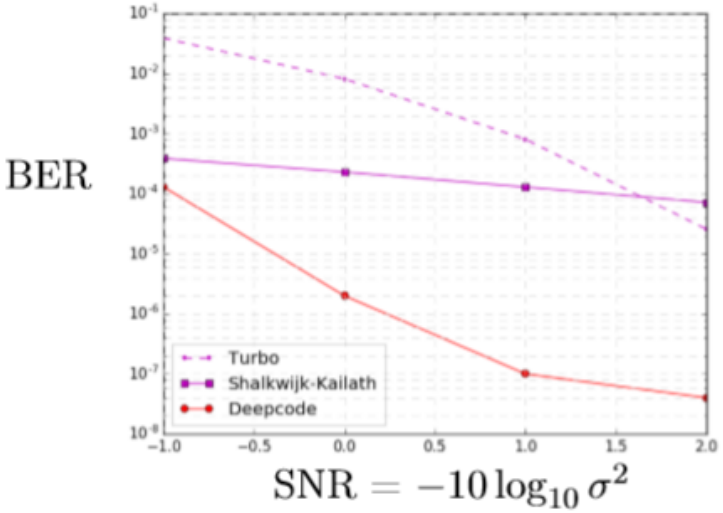
Deepcode / LEARN Codes

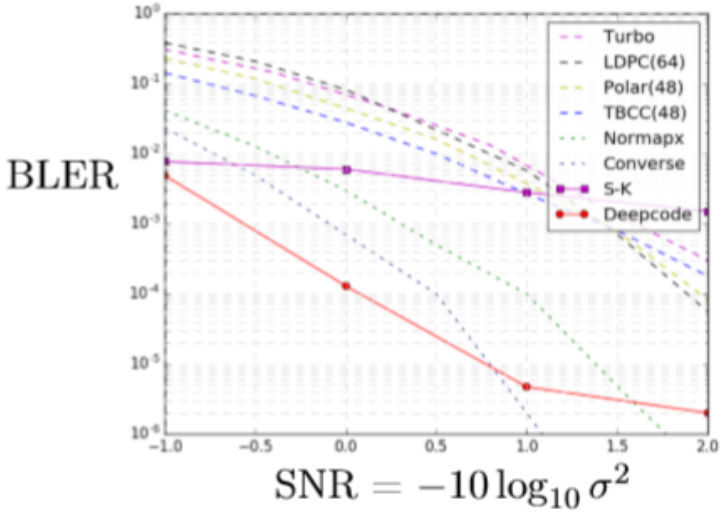
Deepcode: Feedback Codes via Deep Learning

Hyeji Kim*, Yihan Jiang†, Sreeram Kannan†, Sewoong Oh*, Pramod Viswanath*
University of Illinois at Urbana Champaign*, University of Washington†

Encoder A: RNN feedback encoder

- Использование обратной связи
- RNN codes



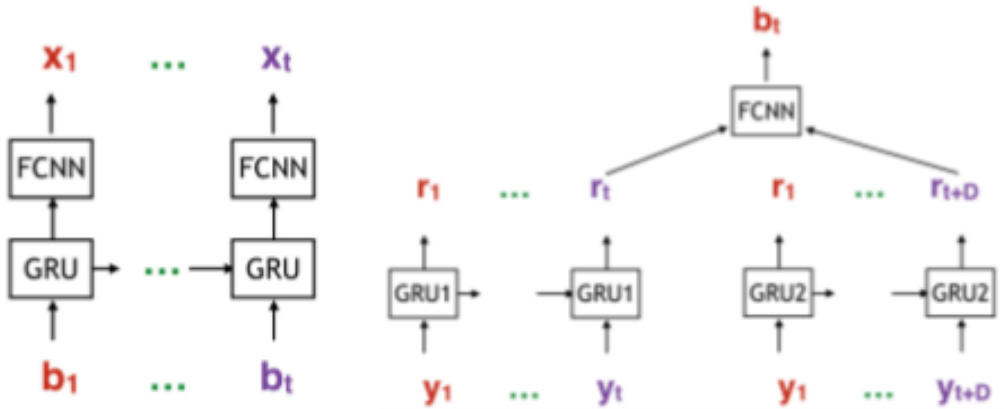


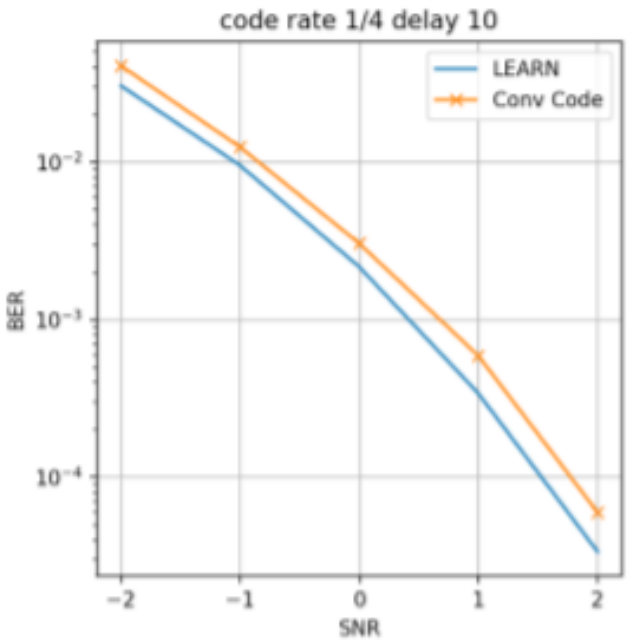
Deepcode / LEARN Codes

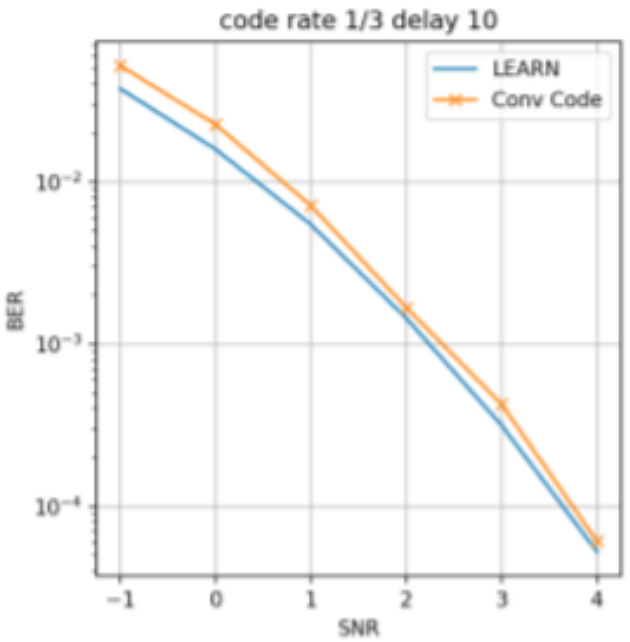
LEARN Codes: Inventing Low-Latency Codes via Recurrent Neural Networks

Yihan Jiang, Hyeji Kim, Himanshu Asnani, Sreeram Kannan, Sewoong Oh, Pramod Viswanath

- Delay
- Превзошли convolutional codes







- Определения
- Введение
- **Задача**
- Решение
- Результаты
- Выводы

Задача

- Построение системы кодирования на нейронных сетях, которая бы отличалась от кода с повторениями

- Определения
- Введение
- Задача
- Решение
- Результаты
- Выводы

Encoder 1 / 2

```
Layer (type)                 Output Shape
=================================================
input_1 (InputLayer)         [(None, 50, 1)]
_________________________________________________
enc_gru_1 (GRU)              (None, 50, 25)
_________________________________________________
enc_gru_2 (GRU)              (None, 50, 25)
_________________________________________________
dense (Dense)                (None, 50, 2)
_________________________________________________
layer_normalization (LayerNo (None, 50, 2)
=================================================
Total params: 6,252
Trainable params: 6,052
Non-trainable params: 200
```

```
Layer (type)                 Output Shape
=================================================
input_4 (InputLayer)         [(None, 10, 1)]
_________________________________________________
enc_gru_1 (GRU)              (None, 10, 25)
_________________________________________________
batch_normalization_1 (Batch (None, 10, 25)
_________________________________________________
enc_gru_2 (GRU)              (None, 10, 25)
_________________________________________________
batch_normalization_2 (Batch (None, 10, 25)
_________________________________________________
time_distributed_2 (TimeDist (None, 10, 2)
_________________________________________________
layer_normalization_1 (Layer (None, 10, 2)
=================================================
Total params: 6,292
Trainable params: 6,152
Non-trainable params: 140
```

Noise 1 / 2

```
_________________________________________________________________
Layer (type)                 Output Shape
=================================================================
input_2 (InputLayer)         [(None, 50, 2)]
_________________________________________________________________
gaussian_noise (GaussianNois (None, 50, 2)
=================================================================
Total params: 0
Trainable params: 0
Non-trainable params: 0
```

```
_________________________________________________________________
Layer (type)                 Output Shape
=================================================================
input_5 (InputLayer)         [(None, 10, 2)]
_________________________________________________________________
gaussian_noise_1 (GaussianNo (None, 10, 2)
=================================================================
Total params: 0
Trainable params: 0
Non-trainable params: 0
```

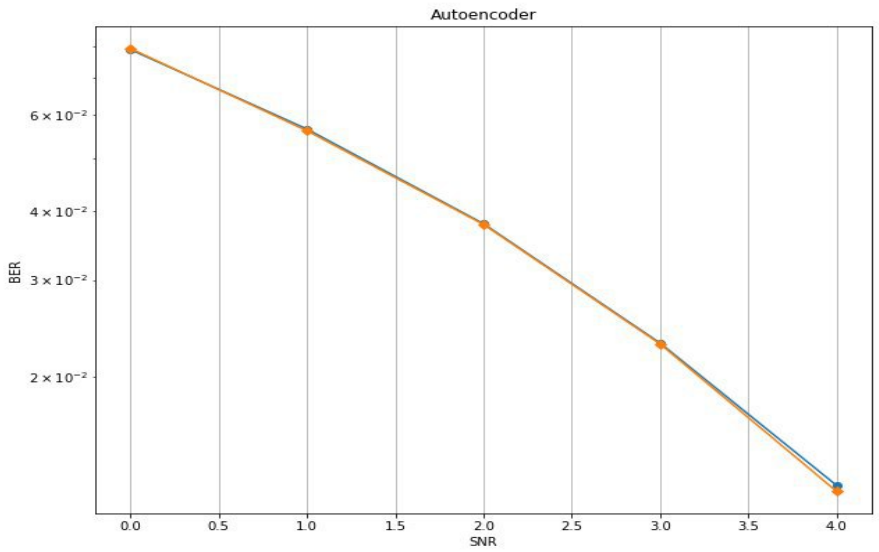
Decoder 1 / 2

```
Layer (type)
==============================
input_3 (InputLayer)
------------------------------
dec_gru_11 (GRU)
------------------------------
dec_gru_21 (GRU)
------------------------------
dec_gru_12 (GRU)
------------------------------
dec_gru_22 (GRU)
------------------------------
concatenate (Concatenate)
------------------------------
dense_1 (Dense)
==============================
```

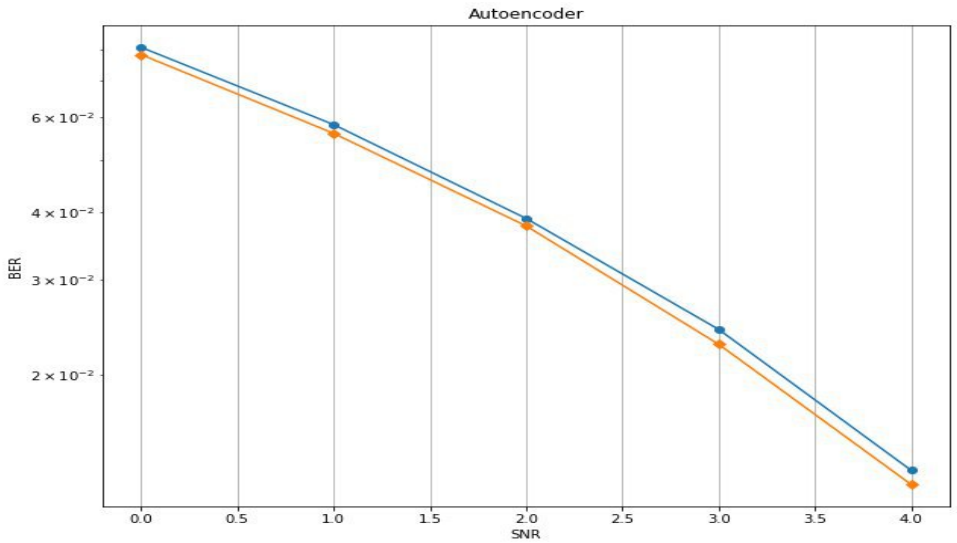
```
Layer (type)
==================================
input_6 (InputLayer)
----------------------------------
batch_normalization_5 (BatchNor
----------------------------------
batch_normalization_3 (BatchNor
----------------------------------
dec_gru_21 (GRU)
----------------------------------
dec_gru_11 (GRU)
----------------------------------
batch_normalization_6 (BatchNor
----------------------------------
batch_normalization_4 (BatchNor
----------------------------------
dec_gru_22 (GRU)
----------------------------------
dec_gru_12 (GRU)
----------------------------------
lambda_1 (Lambda)
----------------------------------
concatenate_1 (Concatenate)
----------------------------------
dense_5 (Dense)
----------------------------------
time_distributed_3 (TimeDistrib
==================================
```

- Определения
- Введение
- Задача
- Решение
- Результаты
- Выводы

1. BER(SNR)

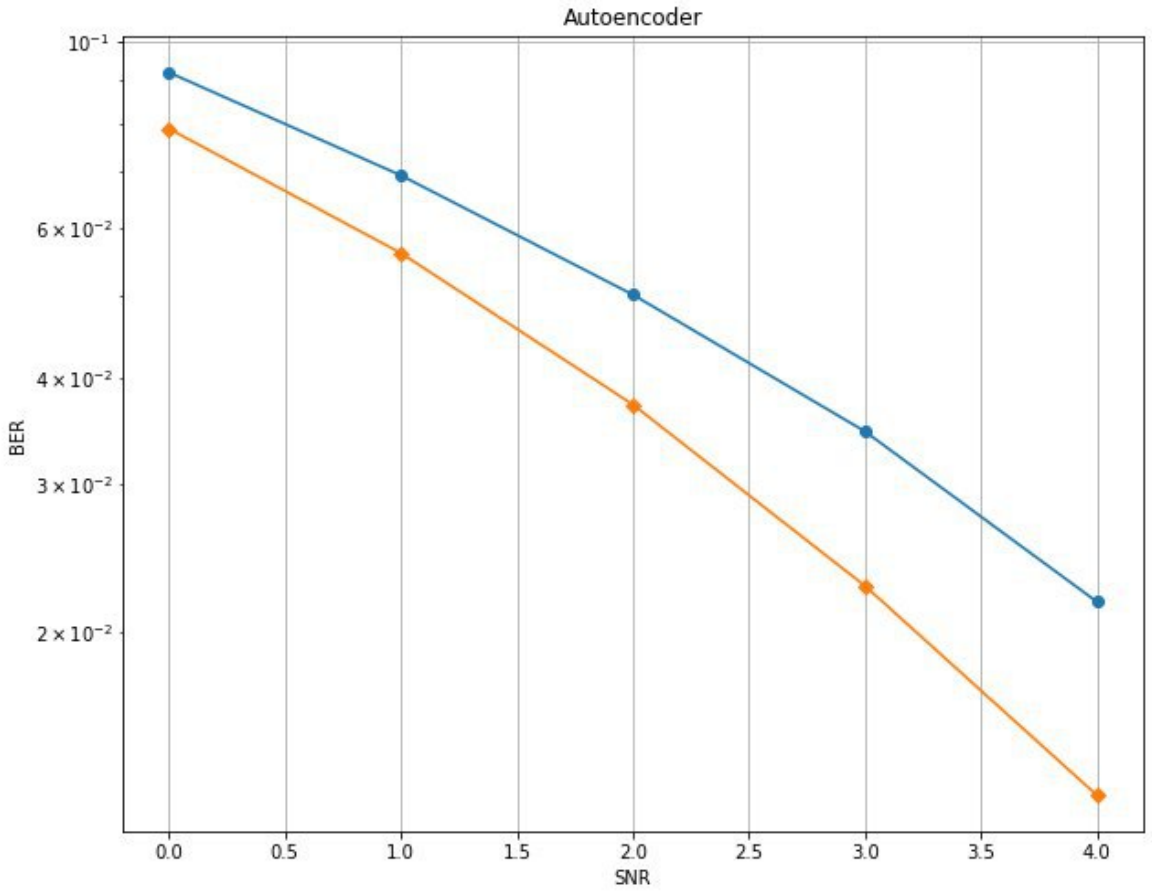


K = 20

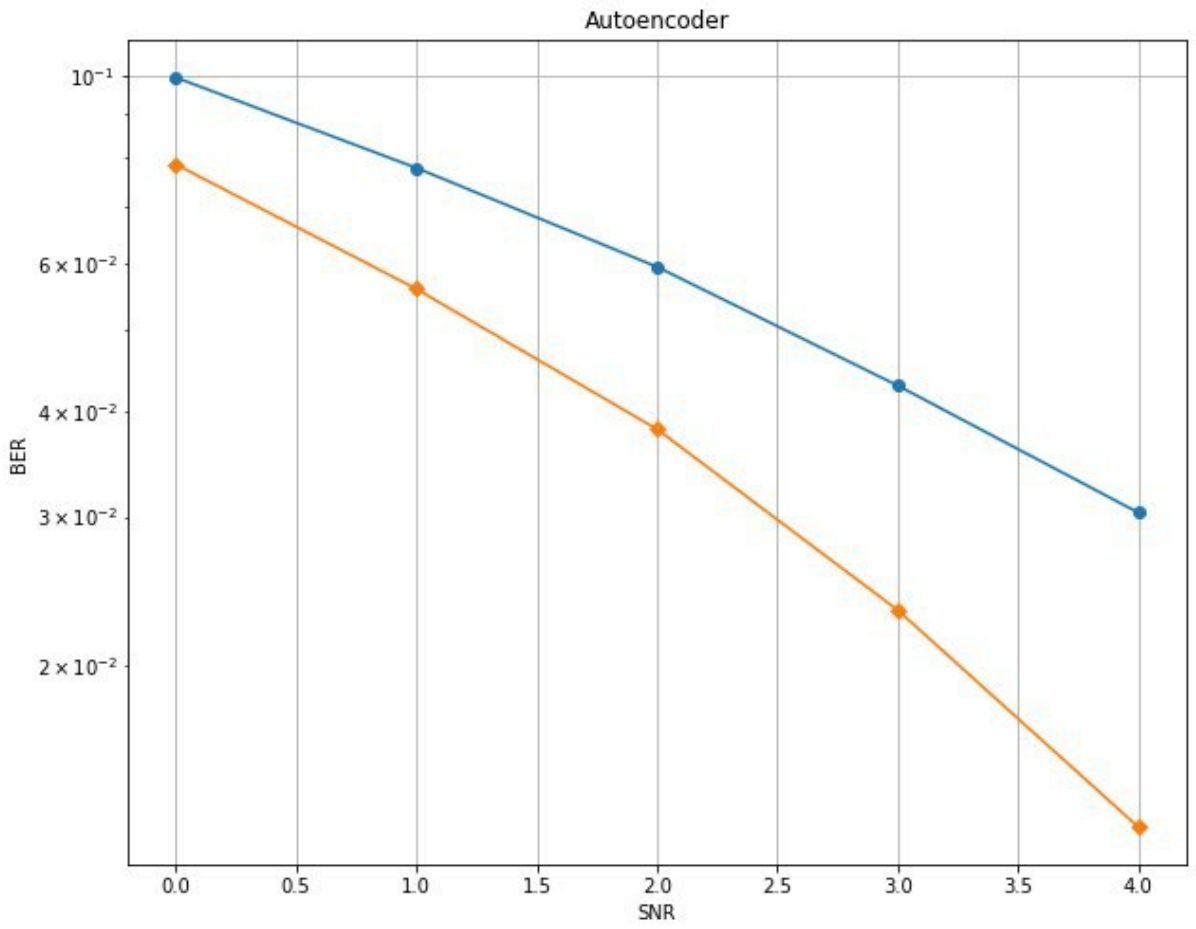


K = 50

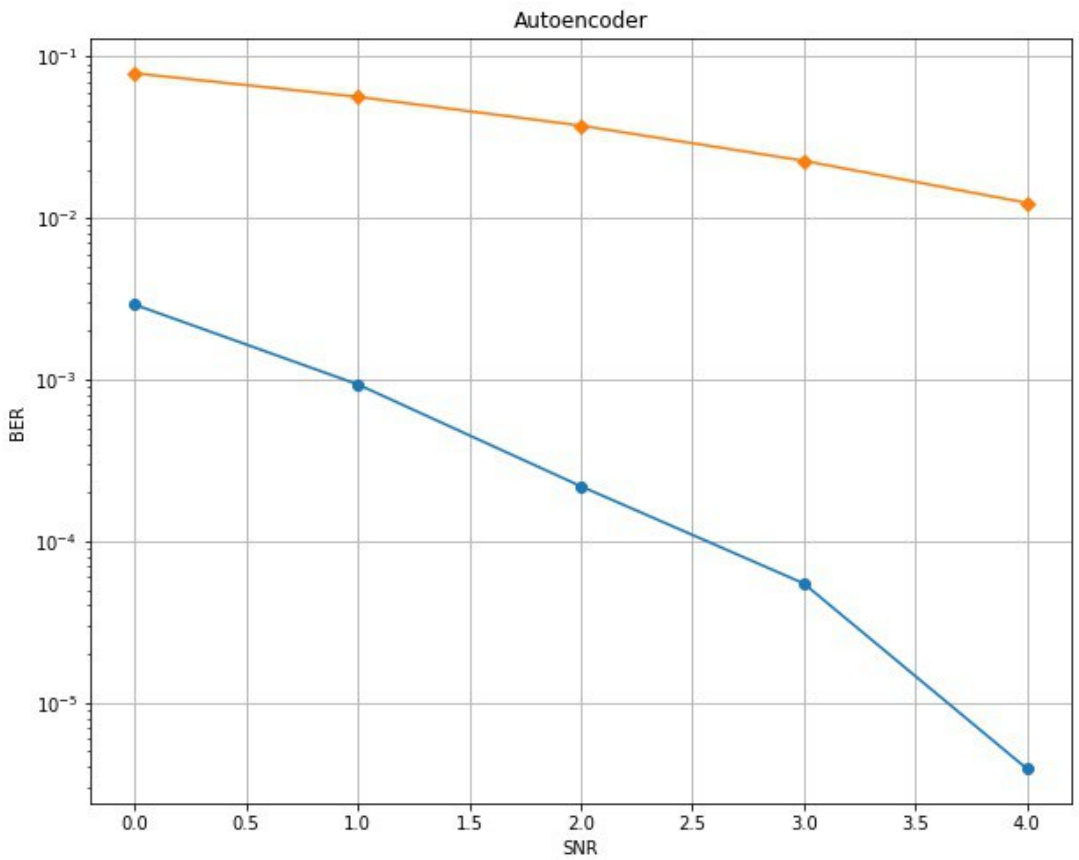
2. BER(SNR)


Autoencoder
10^{-1}
6×10^{-2}
4×10^{-2}
3×10^{-2}
2×10^{-2}
BER
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
SNR

*3. Separate encode and decoder



5. Good results




```
Model: "encoder"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_61 (InputLayer)        [(None, 10, 1)]           0
_________________________________________________________________
enc_gru_1 (GRU)              (None, 10, 1)             12
_________________________________________________________________
batch_normalization_1 (Batch (None, 10, 1)             4
_________________________________________________________________
enc_gru_2 (GRU)              (None, 10, 1)             12
_________________________________________________________________
time_distributed_38 (TimeDis (None, 10, 2)             4
=================================================================
Total params: 32
Trainable params: 30
Non-trainable params: 2
_________________________________________________________________
Model: "noise"
```

```
Model: "decoder"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_63 (InputLayer)           [(None, 10, 2)]      0
__________________________________________________________________________________________________
dec_gru_12 (GRU)                (None, 10, 1)        15          input_63[0][0]
__________________________________________________________________________________________________
dec_gru_11 (GRU)                (None, 10, 1)        15          input_63[0][0]
__________________________________________________________________________________________________
lambda_16 (Lambda)              (None, 10, 1)        0           dec_gru_12[0][0]
__________________________________________________________________________________________________
linear_16 (Linear)              (None, 10, 1)        2           dec_gru_11[0][0]
                                                                 lambda_16[0][0]
__________________________________________________________________________________________________
time_distributed_39 (TimeDistri (None, 10, 1)        2           linear_16[0][0]
==================================================================================================
Total params: 34
Trainable params: 34
Non-trainable params: 0
```

- Определения
- Введение
- Задача
- Решение
- Результаты
- Выводы

Выводы

- Получили код с повторениями
- Исследовали поведение сети для разных длин кодов
- Воспроизвели архитектуру, описанную в статье

Outline

- Применить регуляризацию со штрафами за код с повторениями
- Изучить другие архитектуры нейронных сетей

Team

- Четвериков Илья

Архитектура соединенных энкодера и декодера

- Горбунов Иван

Настройка алгоритма обучения нейронной сети

- Андрианова Екатерина

Исследование архитектур на коротких и длинных кодах

- Котельникова Ольга

Изучение теории, анализ результатов, поиск новых архитектур

Спасибо за внимание