OTUS

ОНЛАЙН-ОБРАЗОВАНИЕ

Рекуррентные сети(и не только)

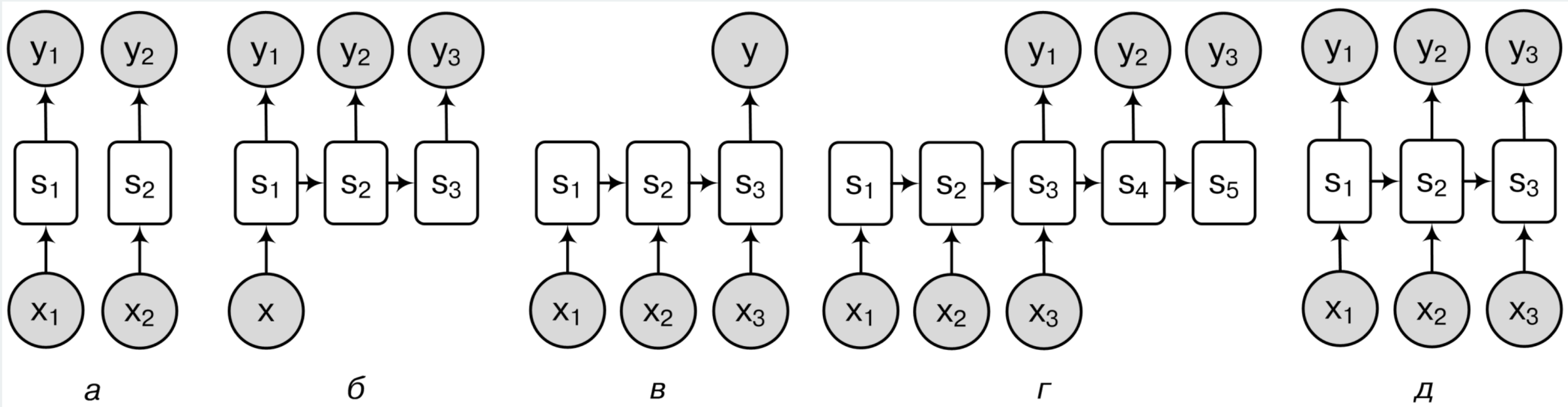
Будущее должно быть заложено в настоящем

Артур Кадурин
Преподаватель

План на сегодня

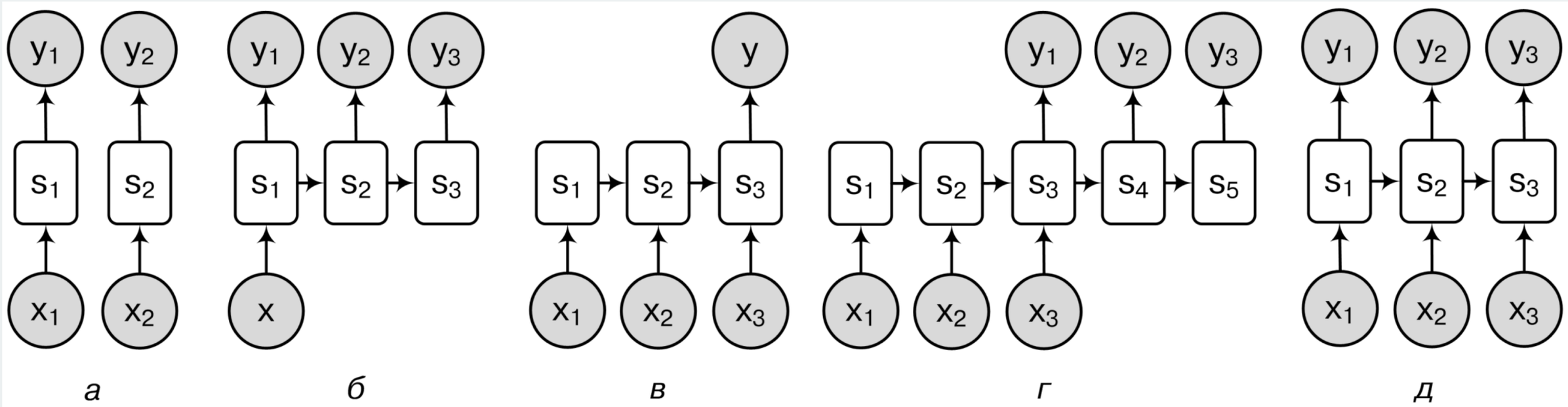
1. **Последовательности: задачи и данные**
2. Рекуррентные нейронные сети
3. Ячейки, слои и свойства
4. BatchNorm для рекуррентных сетей
5. Практика: Wikipedia

Обработка последовательностей



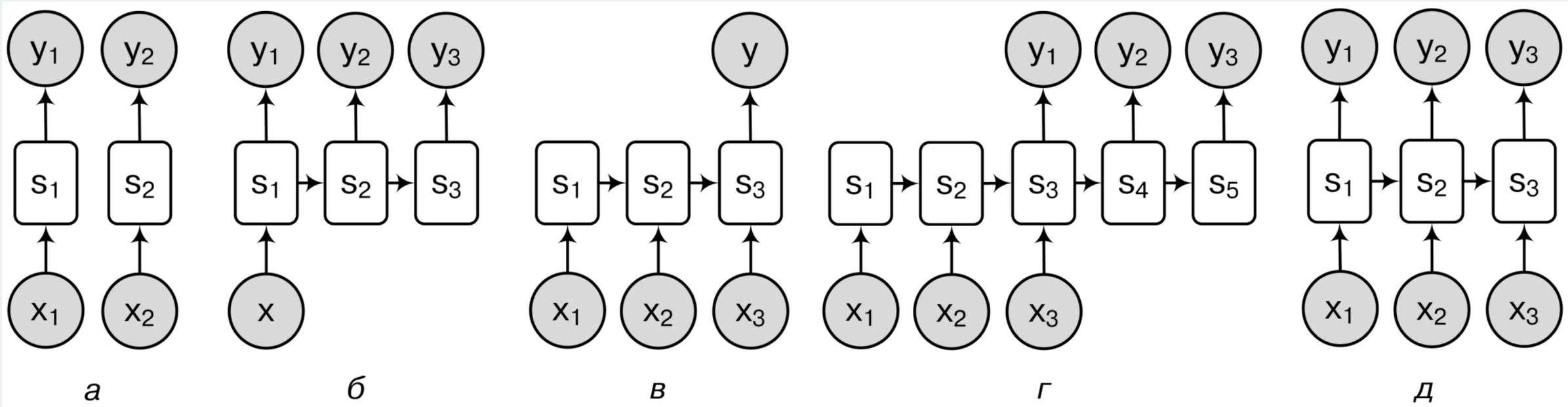
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
Любая задача может быть сведена к этой!

Обработка последовательностей



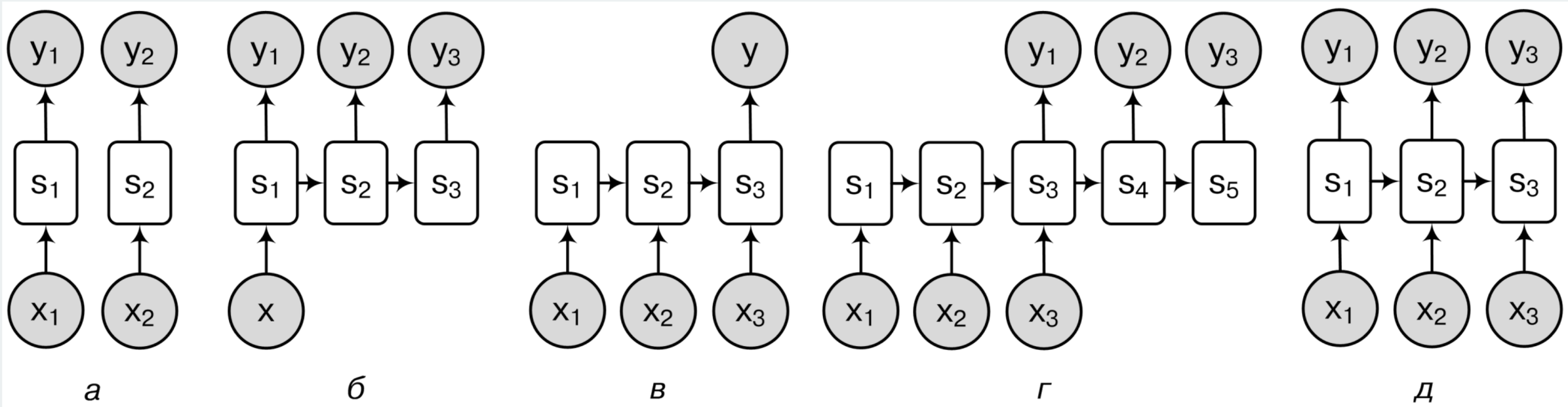
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример?

Обработка последовательностей



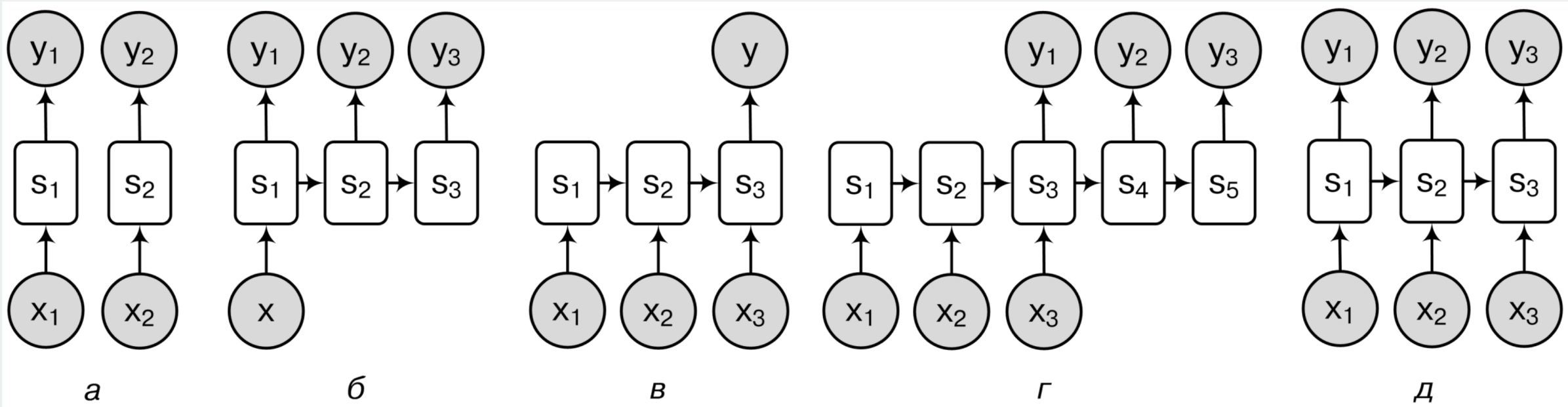
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: **аннотация изображения**

Обработка последовательностей



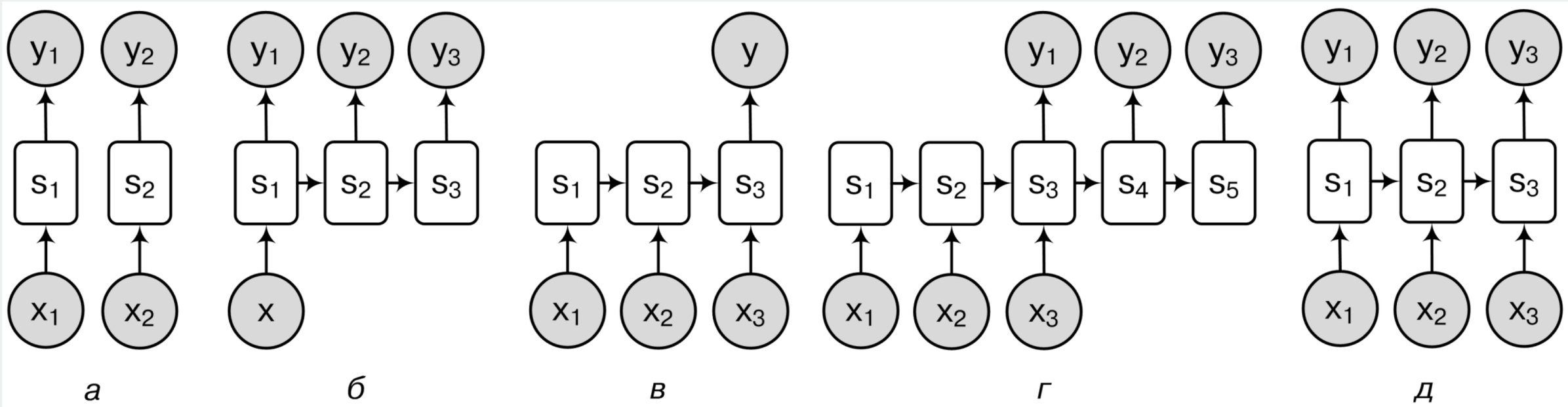
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример?

Обработка последовательностей



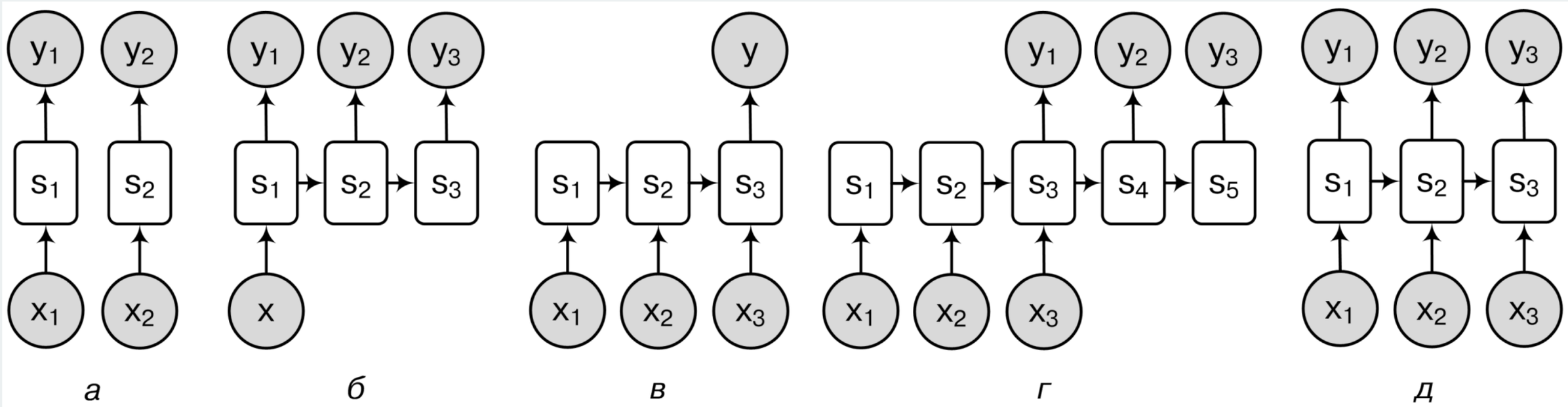
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример: **классификация**

Обработка последовательностей



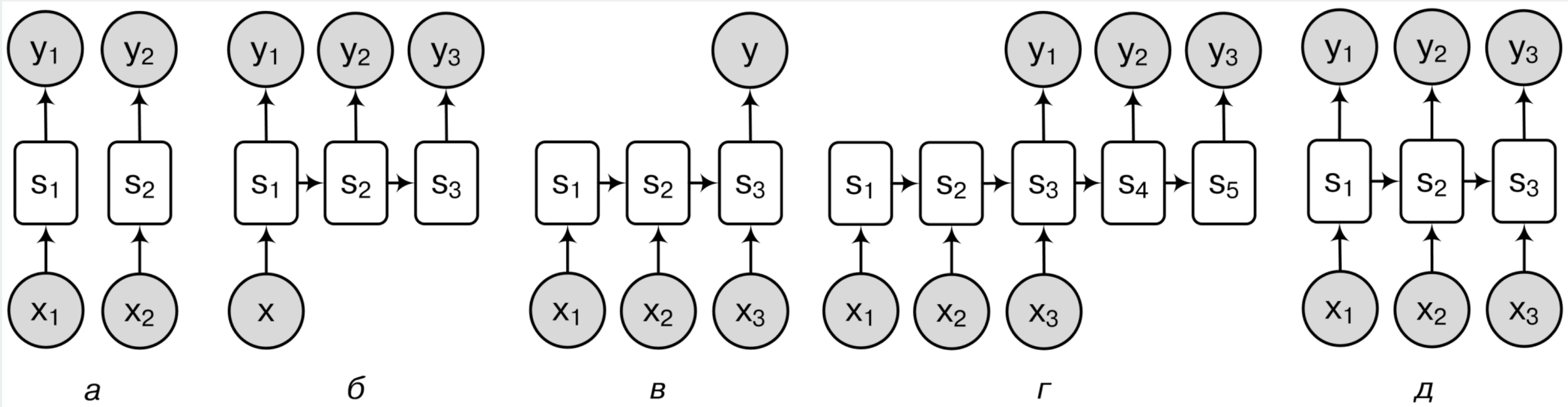
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример: классификация
г. Последовательность входов — последовательность выходов. Пример?

Обработка последовательностей



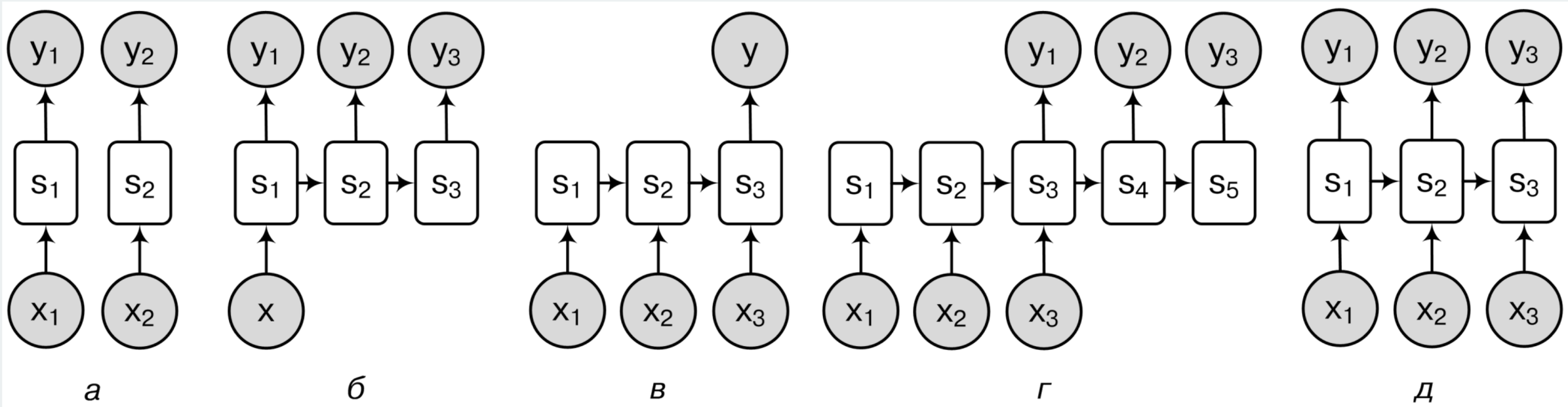
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример: классификация
г. Последовательность входов — последовательность выходов. Пример: **перевод**

Обработка последовательностей



Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример: классификация
г. Последовательность входов — последовательность выходов. Пример: перевод
д. Синхронизированные последовательности. Пример?

Обработка последовательностей



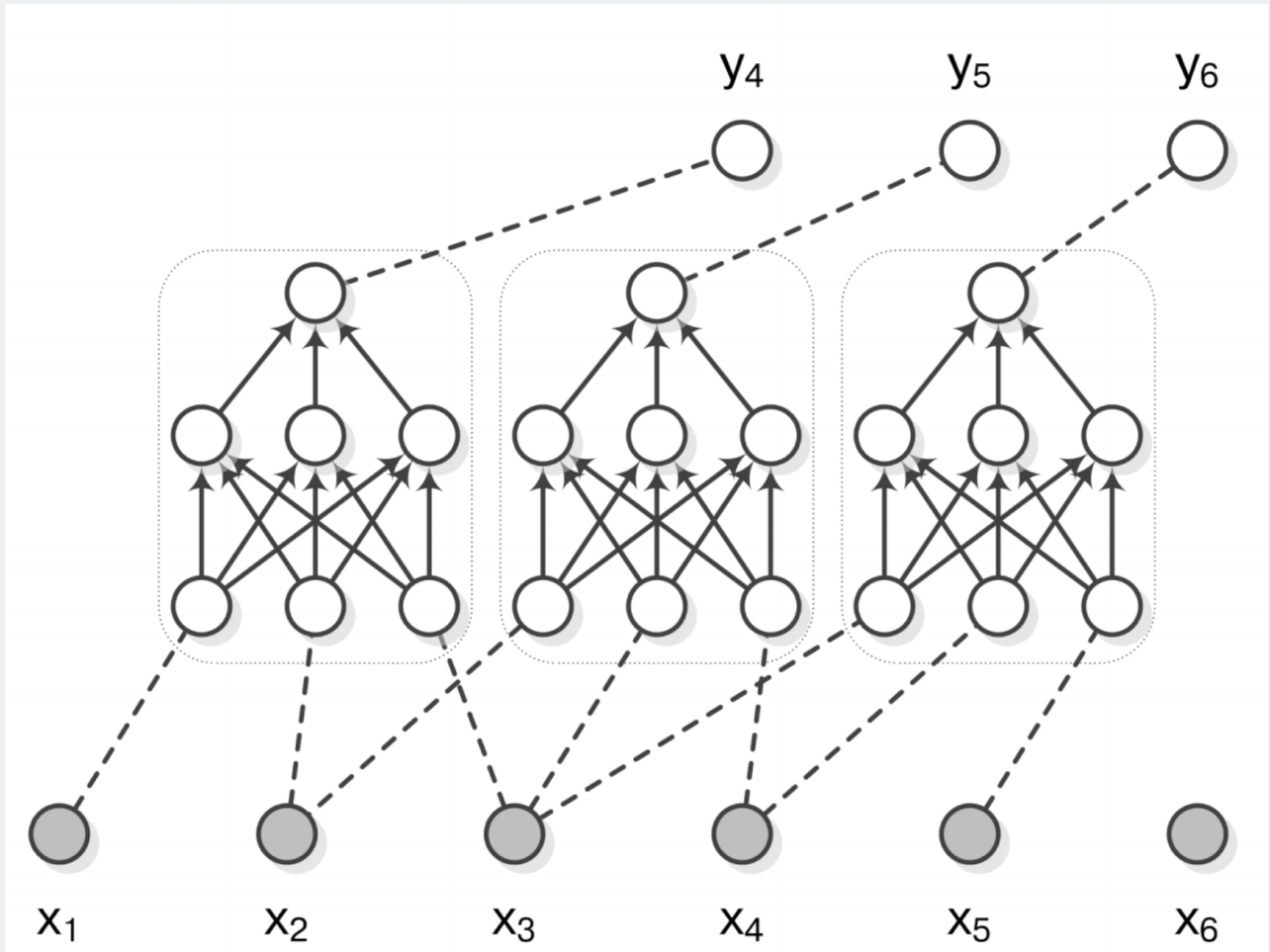
Задачи обработки последовательностей можно разделить на 5 видов
а. Один вход — один выход. Обычная нейросеть.
б. Один вход — последовательность выходов. Пример: аннотация изображения
в. Последовательность входов — один выход. Пример: классификация
г. Последовательность входов — последовательность выходов. Пример: перевод
д. Синхронизированные последовательности. Пример: **Аннотация видео**

Простое решение

Зафиксировав длину последовательности мы можем предсказывать ее свойства или очередной элемент с помощью многослойной полносвязной нейросети.
Какие проблемы?

Простое решение

Зафиксировав длину последовательности мы можем предсказывать ее свойства или очередной элемент с помощью многослойной полносвязной нейросети.
Можно делать свертки по времени!



Артур Кадурин, преподаватель

Представление данных

Если корпус текстов содержит ограниченное количество слов, то каждое слово может быть представлено **one-hot** вектором размерности равной количеству различных слов в корпусе. Для больших словарей проще хранить **id** слова.
Для этого в PyTorch есть слой nn.Embeddings
Проблемы?

Представление данных

Если корпус текстов содержит ограниченное количество слов, то каждое слово может быть представлено **one-hot** вектором размерности равной количеству различных слов в корпусе. Для больших словарей проще хранить **id** слова.

1. Слов очень много $\sim 10^6$
2. Словоформ очень много(особенно в русском языке)
3. Редкие слова — очень редкие.

Представление данных

Если корпус текстов содержит ограниченное количество слов, то каждое слово может быть представлено **one-hot** вектором размерности равной количеству различных слов в корпусе. Для больших словарей проще хранить **id** слова.

Слово может быть представлено списком **n-грамм**:

#слово# —> [#сл, сло, лов, ово, во#]

Преимущества?

Представление данных

Если корпус текстов содержит ограниченное количество слов, то каждое слово может быть представлено **one-hot** вектором размерности равной количеству различных слов в корпусе. Для больших словарей проще хранить **id** слова.

Слово может быть представлено списком **n-грамм**:

#слово# —> [#сл, сло, лов, ово, во#]

1. Словарь **n-грамм** почти всегда сильно меньше обычного. $\sim 10^4$
2. Словоформы похожи!
3. Редкие слова сразу! имеют смысл

Проблемы?

Представление данных

Если корпус текстов содержит ограниченное количество слов, то каждое слово может быть представлено **one-hot** вектором размерности равной количеству различных слов в корпусе. Для больших словарей проще хранить **id** слова.

Слово может быть представлено списком **n-грамм**:

#слово# —> [#сл, сло, лов, ово, во#]

1. Словарь **n-грамм** почти всегда сильно меньше обычного. $\sim 10^4$
2. Словоформы похожи!
3. Редкие слова сразу! имеют смысл

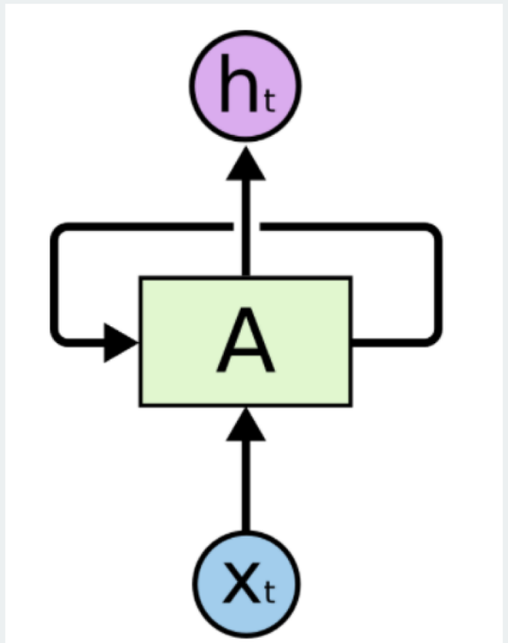
Проблемы:

1. Разное количество токенов в слове —> разный уровень активации в первом слое
2. Коллизии. (на самом деле их очень мало)

План на сегодня

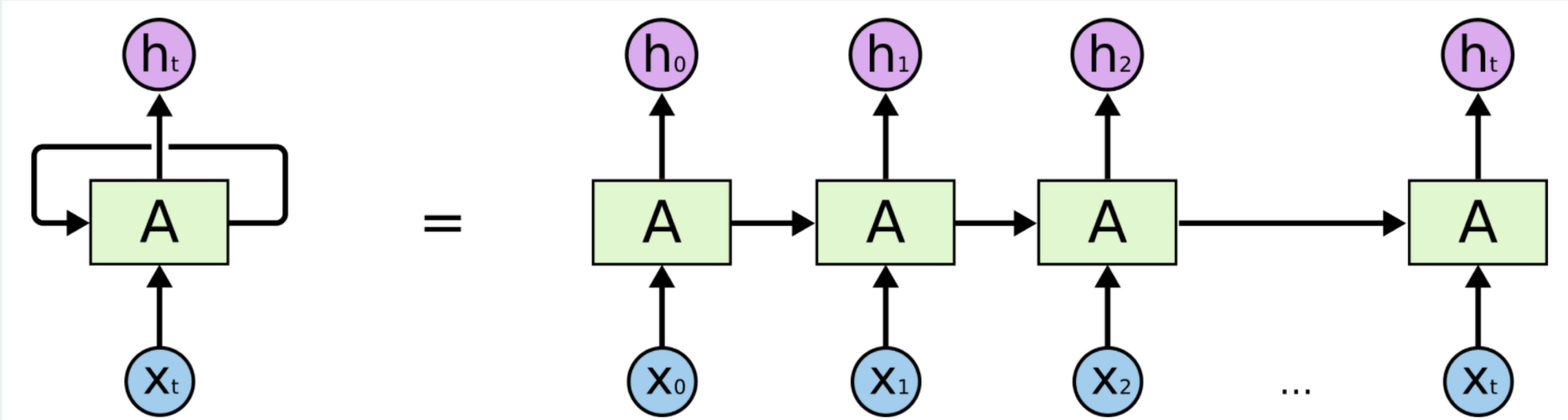
1. Последовательности: задачи и данные
2. **Рекуррентные нейронные сети**
3. Ячейки, слои и свойства
4. BatchNorm для рекуррентных сетей
5. Практика: Wikipedia

Рекуррентная нейронная сеть



A — ячейка рекуррентной нейронной сети, она получает на вход очередной элемент последовательности вместе с выходом с предыдущего шага и возвращает какой-то результат.

Рекуррентная нейронная сеть

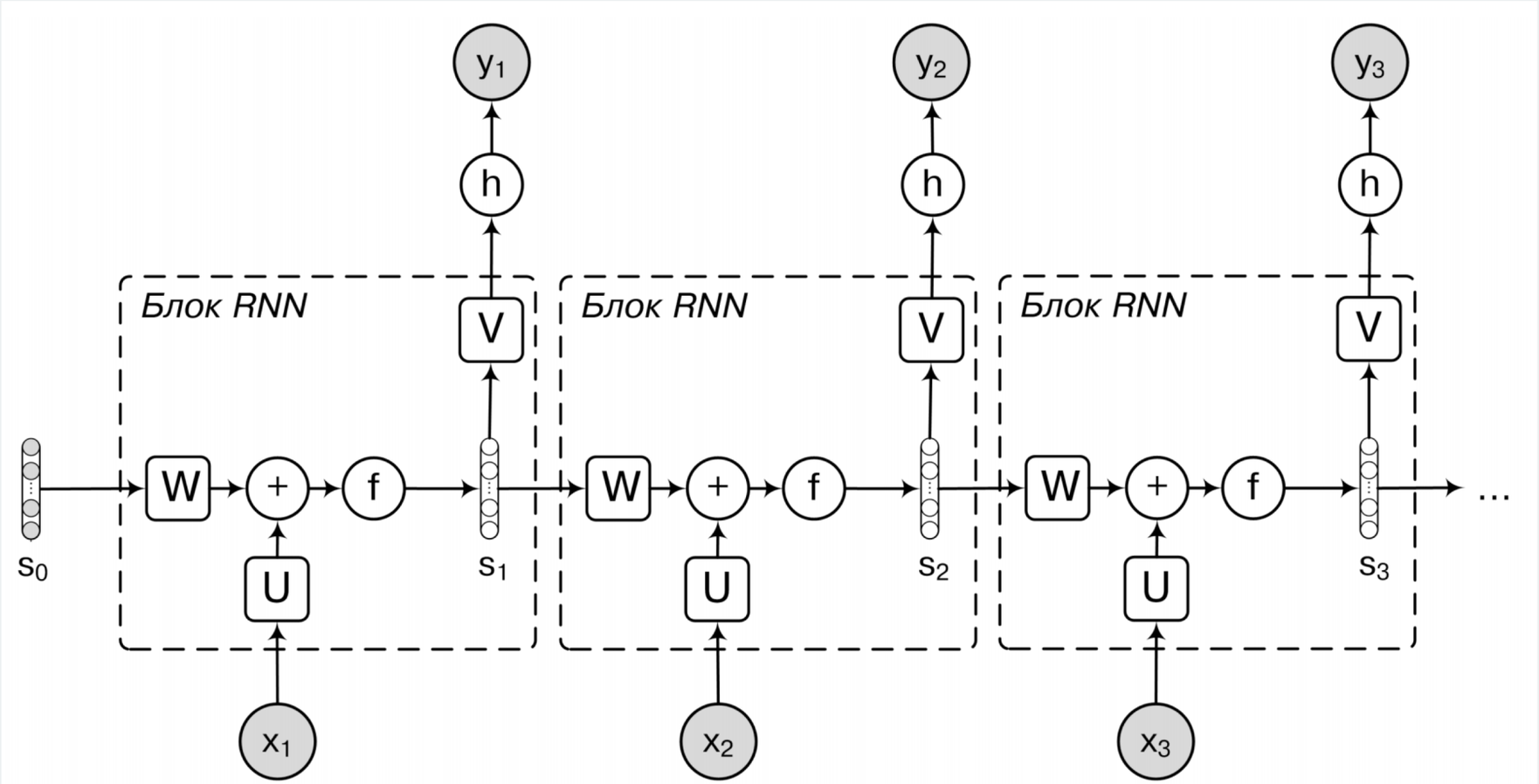


A — ячейка рекуррентной нейронной сети, она получает на вход очередной элемент последовательности вместе с выходом с предыдущего шага и возвращает какой-то результат.
Рекуррентная нейронная сеть подразумевает циклы. Но на самом деле, когда мы говорим про обработку последовательностей, мы можем развернуть цикл в последовательные вычисления и применять стандартные методы обучения.

План на сегодня

1. Последовательности: задачи и данные
2. Рекуррентные нейронные сети
3. **Ячейки, слои и свойства**
4. BatchNorm для рекуррентных сетей
5. Практика: Wikipedia

Простейшая рекуррентная ячейка

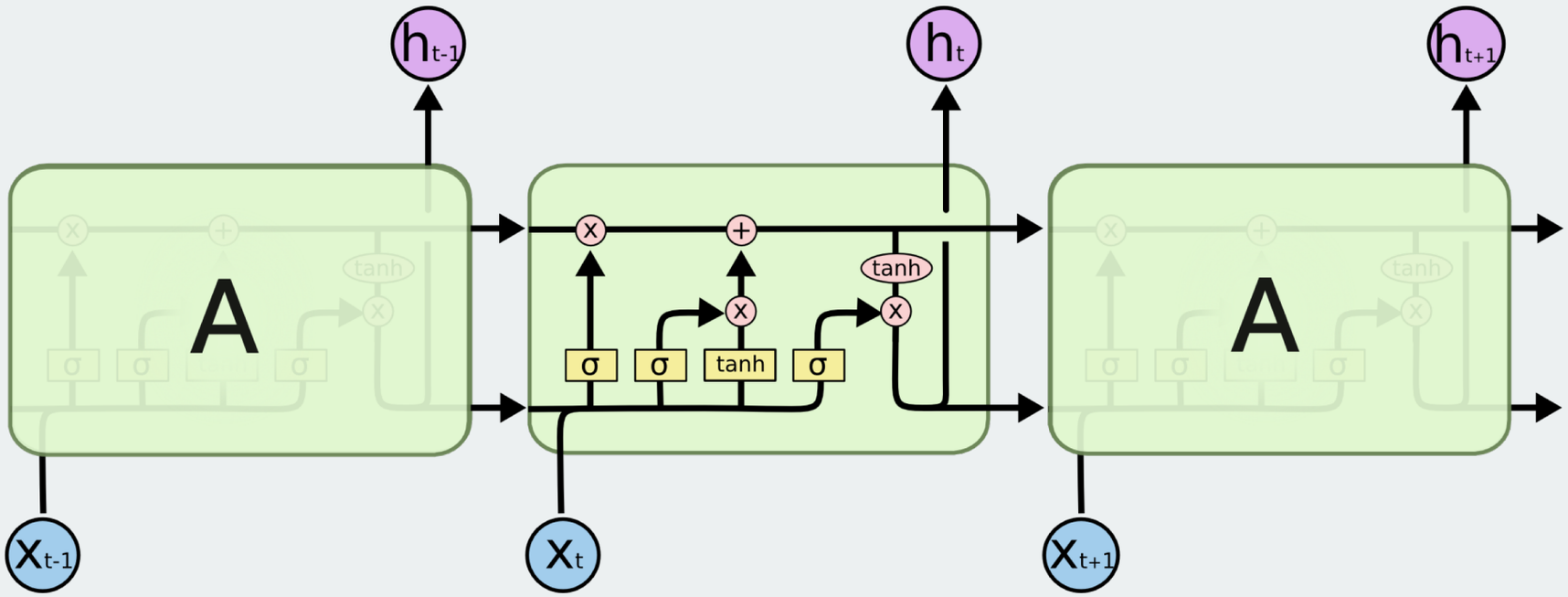


$$\boldsymbol{a_t} = \boldsymbol{b} + W\boldsymbol{s_{t-1}} + U\boldsymbol{x_t}, \qquad \boldsymbol{s_t} = f(\boldsymbol{a_t})$$

$$\boldsymbol{o_t} = \boldsymbol{c} + V\boldsymbol{s_t}, \qquad \boldsymbol{y_t} = h(\boldsymbol{o_t})$$

Артур Кадурин, преподаватель

Long Short-Term Memory



LSTM-ячейка устроена намного сложнее. Она содержит 3 основных вида гейтов:
1. **Забывающий гейт** «чистит» память.
2. **Входной гейт** выбирает какую новую информацию сохранить в ячейке.
3. **Выходной гейт** выбирает что ячейка выдаст в качестве выходов.

$$\boldsymbol{c'_t} = tanh(W_{xc}\boldsymbol{x_t} + W_{hc}\boldsymbol{h_{t-1}} + \boldsymbol{b_{c'}})$$

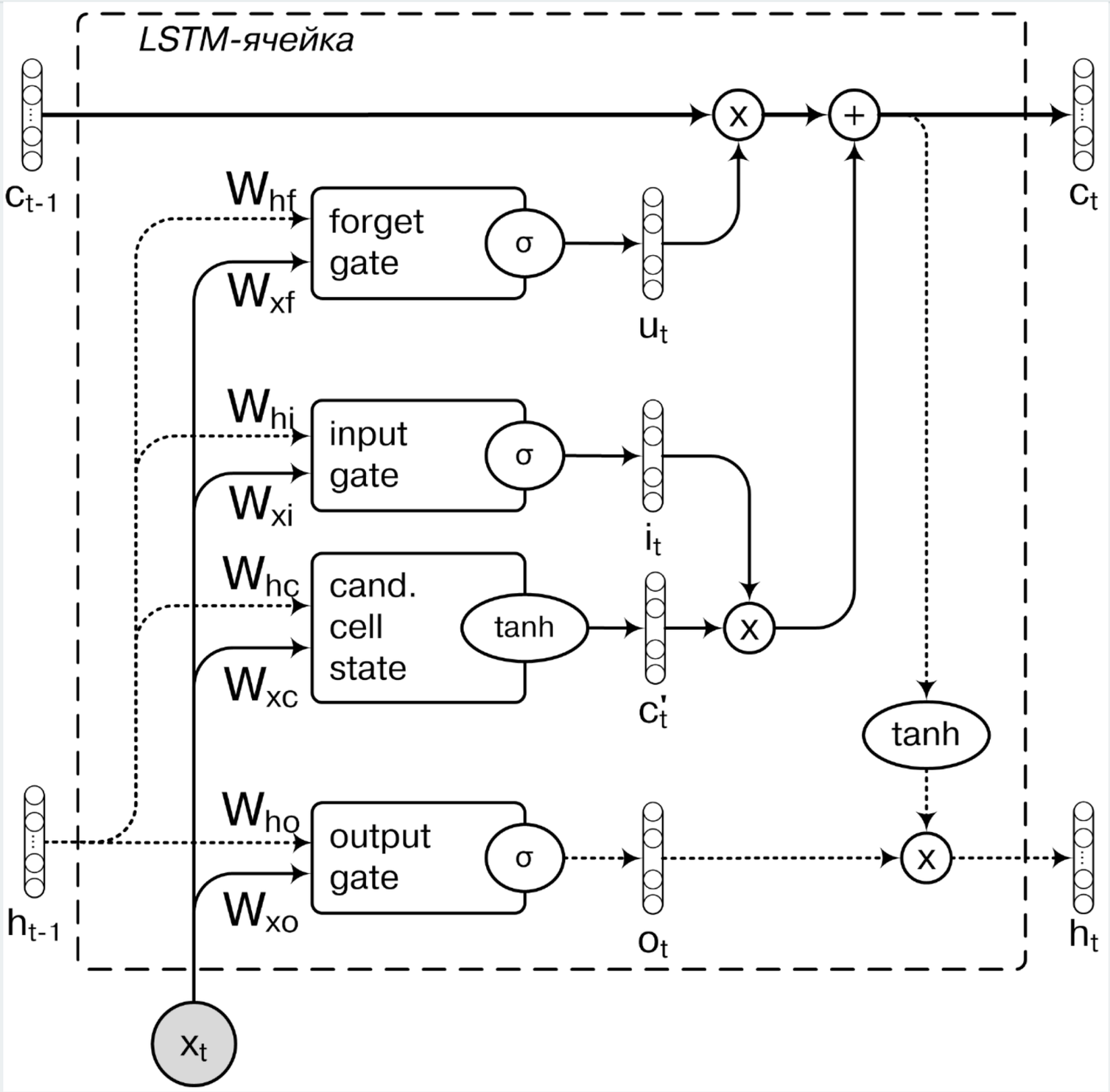
$$\boldsymbol{i_t} = \sigma(W_{xi}\boldsymbol{x_t} + W_{hi}\boldsymbol{h_{t-1}} + \boldsymbol{b_i})$$

$$\boldsymbol{f_t} = \sigma\left(W_{xf}\boldsymbol{x_t} + W_{hf}\boldsymbol{h_{t-1}} + \boldsymbol{b_f}\right)$$

$$\boldsymbol{o_t} = \sigma(W_{xo}\boldsymbol{x_t} + W_{ho}\boldsymbol{h_{t-1}} + \boldsymbol{b_o})$$

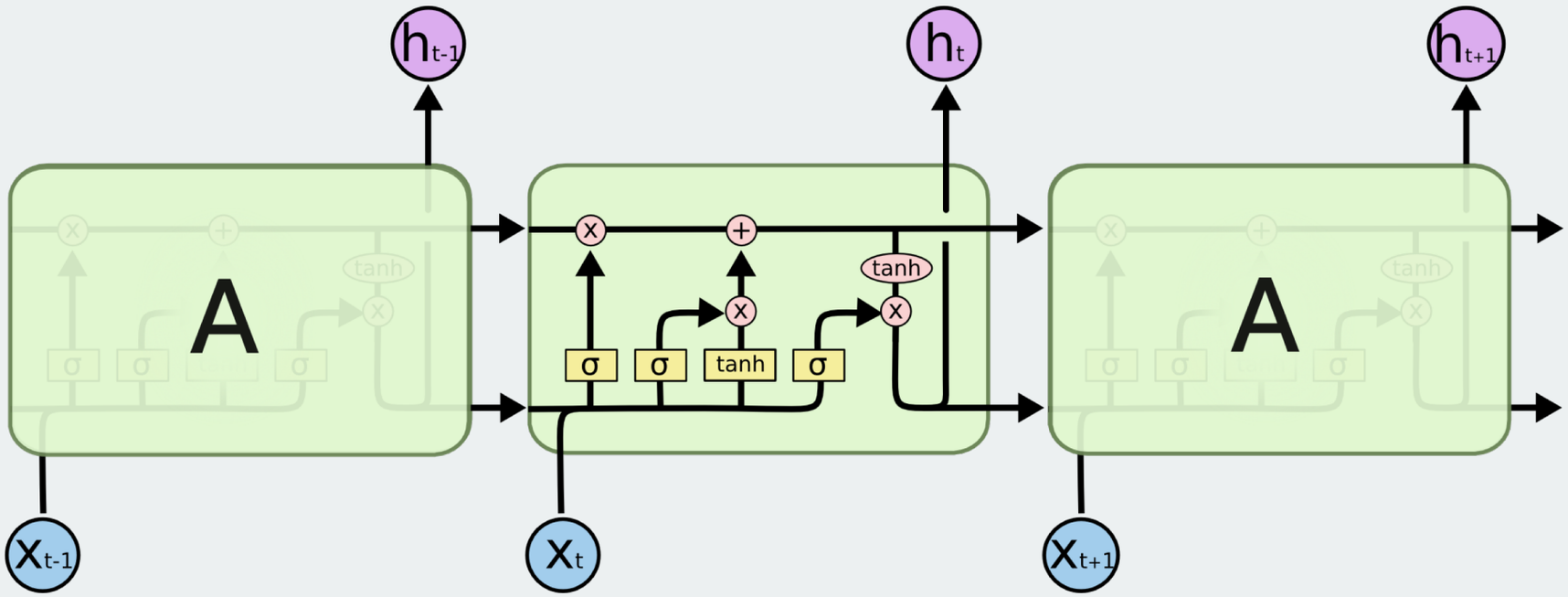
$$\boldsymbol{c_t} = \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \boldsymbol{c'_t}$$

$$\boldsymbol{h_t} = \boldsymbol{o_t} \odot tanh(\boldsymbol{c_t})$$



Артур Кадурин, преподаватель

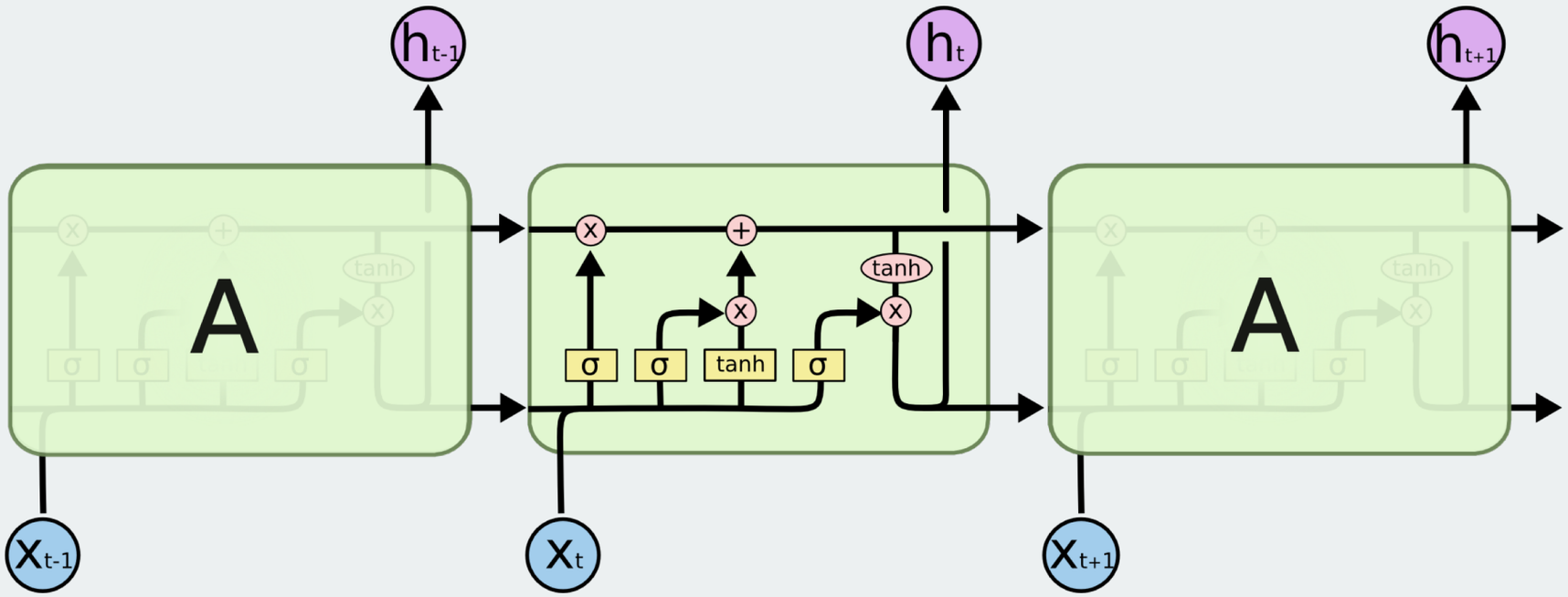
Карусель константной ошибки



$$c_t = f_t \odot c_{t-1} + i_t \odot c'_t$$

Артур Кадурин, преподаватель

Карусель константной ошибки

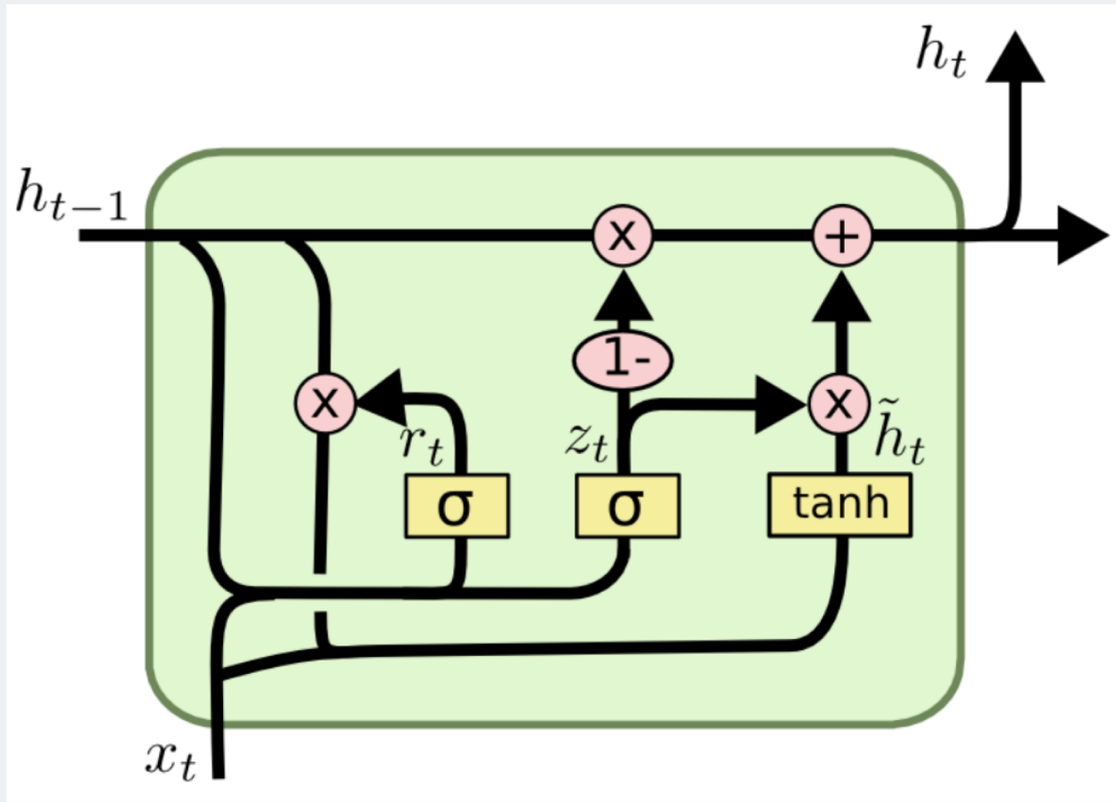


$$\boldsymbol{c_t} = \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \boldsymbol{c'_t}$$

$$\boldsymbol{f_t} = \sigma\left(W_{xf}\boldsymbol{x_t} + W_{hf}\boldsymbol{h_{t-1}} + \boldsymbol{b_f}\right)$$

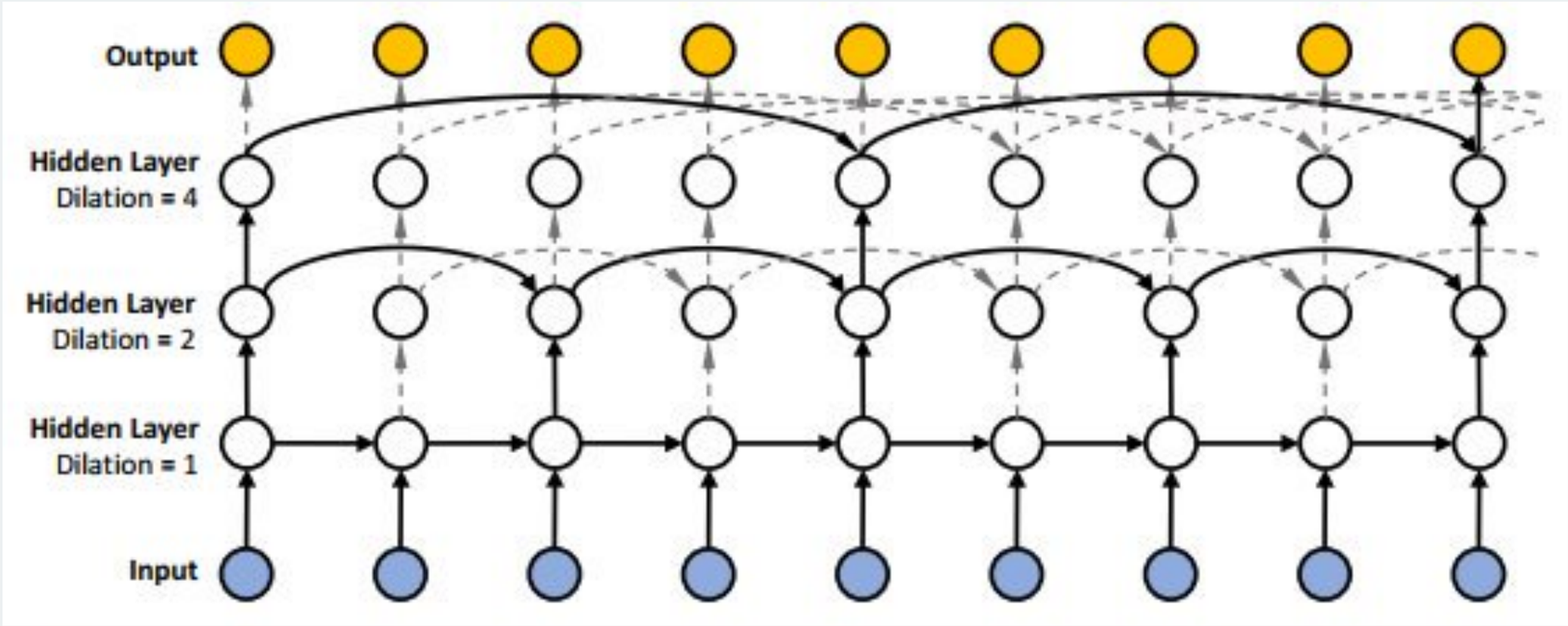
Артур Кадурин, преподаватель

Gated Recurrent Unit



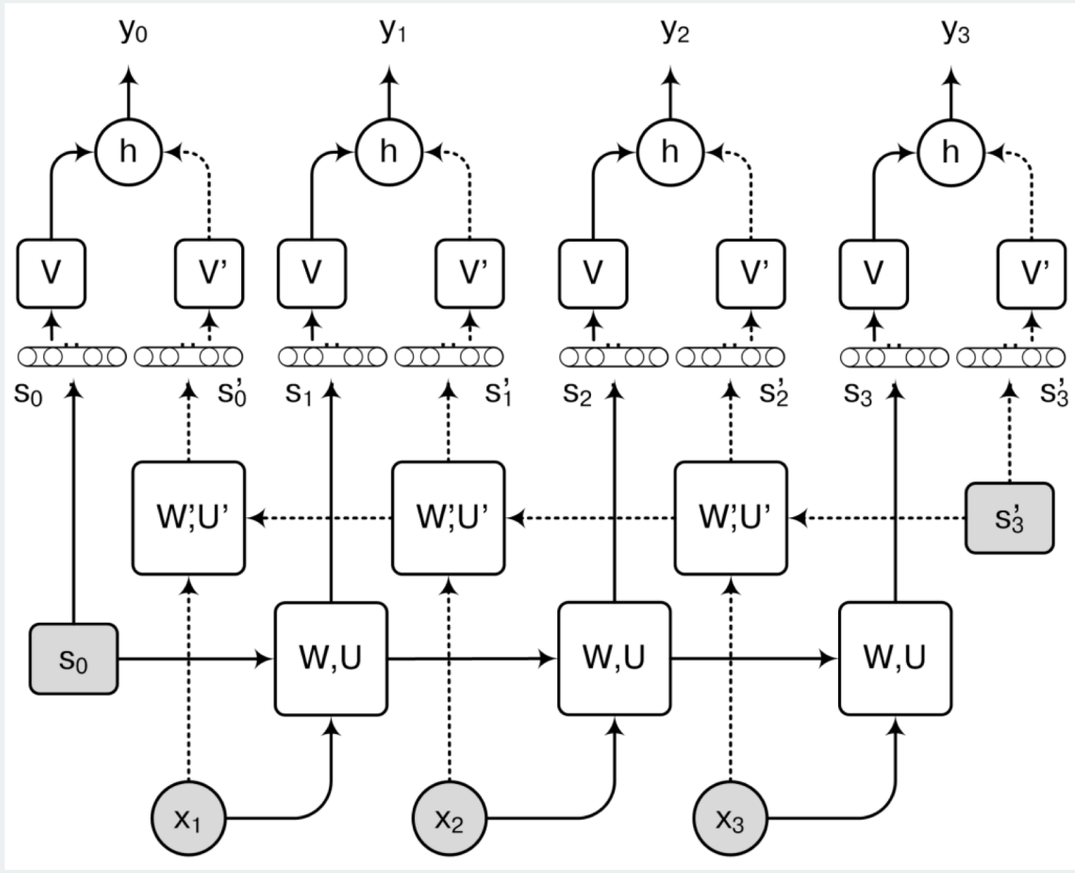
Несмотря на визуально более сложную структуру ячейки **GRU** требует меньшего количества параметров. Кроме того, **GRU** объединяет состояние ячейки и память в один вектор.

Параметры



Так же как к сверточным сетям, к рекуррентным сетям применимы растяжение и шаг.

Параметры



Так же как к сверточным сетям, к рекуррентным сетям применимы растяжение и шаг.

Но еще у них есть направление!

План на сегодня

1. Последовательности: задачи и данные
2. Рекуррентные нейронные сети
3. Ячейки, слои и свойства
4. **BatchNorm для рекуррентных сетей**
5. Практика: Wikipedia

Batch Normalization

$$y_k = \gamma_k \hat{x}_k + \beta_k = \gamma_k \frac{x_k - \mathbb{E}[x_k]}{\sqrt{Var(x_k)}} + \beta_k$$

В чем проблема?

Batch Normalization

$$y_k = \gamma_k \hat{x}_k + \beta_k = \gamma_k \frac{x_k - \mathbb{E}[x_k]}{\sqrt{Var(x_k)}} + \beta_k$$

В чем проблема?

1. Элементы разных последовательностей на разном шаге могут иметь разную семантику. Например часть последовательностей вообще может закончиться.
2. Ячейка — одна, а шагов много, какие веса использовать?

Batch Normalization

$$y_k = \gamma_k \hat{x}_k + \beta_k = \gamma_k \frac{x_k - \mathbb{E}[x_k]}{\sqrt{Var(x_k)}} + \beta_k$$

В чем проблема?

1. Элементы разных последовательностей на разном шаге могут иметь разную семантику. Например часть последовательностей вообще может закончиться.
2. Ячейка — одна, а шагов много, какие веса использовать?

$$\boldsymbol{h_t} = f\left(BN(W_h \boldsymbol{h_{t-1}}; \boldsymbol{\beta_h}, \boldsymbol{\gamma_h}) + BN(W_x \boldsymbol{x_t}; \boldsymbol{\beta_x}, \boldsymbol{\gamma_x})\right)$$

$$\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_T) \quad \boldsymbol{\gamma} = (\gamma_1, \gamma_2, \dots, \gamma_T)$$

Layer Normalization

$$y_k = \gamma \hat{x}_k + \beta = \gamma \frac{x_k - \mathbb{E}[\boldsymbol{x}]}{\sqrt{Var(\boldsymbol{x})}} + \beta$$

А что, если нормировать без привязки к батчу?
Раньше мы брали среднее по нулевой размерности, а теперь будем брать среднее по первой размерности и все — ок!

Weight Normalization

$$y_k = \frac{\gamma}{\|\boldsymbol{w}_k\|}\boldsymbol{w}_k^T\boldsymbol{x} + \boldsymbol{b}_k$$

Но и это еще не все! А давайте вообще нормировать не активации, а веса!
Этот подход, конечно, применим не только к рекуррентым сетям.

План на сегодня

1. Последовательности: задачи и данные
2. Рекуррентные нейронные сети
3. Ячейки, слои и свойства
4. BatchNorm для рекуррентных сетей
5. **Практика: Wikipedia**

Perplexity и Температура

$$2^{\mathrm{H}(P,Q)} = 2^{-\sum_i p_i \log q_i}$$

Перплексити — это мера того, насколько хорошо модель предсказывает данные

$$p(w) \propto e^{-\frac{1}{T}x_w}$$

T — это температура. Так как каждый символ мы сэмплируем из многомерного распределения Бернулли, то добавив параметр T мы можем «сгладить» вероятности или, наоборот, »растащить»

Спасибо
за внимание!