Mamba: Linear-Time Sequence Modeling with Selective State Spaces

Подготовил:
Казадаев Максим, БПМИ202

План

- **SSM (State Space Model)** – повторение
- **Selective State Space Model** – улучшение модели
- **Mamba** – архитектура и интерпретация
- Эксперименты
- Вывод: **когда применять?**

SSM (State Space Model)

Рассмотрим процесс, заданный диффуром:

$$h'(t) = Ah(t) + Bx(t)$$
$$y(t) = Ch(t)$$

x(t) — вход
h(t) — скрытое состояние
y(t) — выход
A, B, C — параметры

Дискретизация диффура с шагом delta: $(\Delta, A, B, C) \mapsto (\overline{A}, \overline{B}, C)$

$$h_t = \overline{A}h_{t-1} + \overline{B}x_t$$
$$y_t = Ch_t$$

$$\overline{A} = \exp(\Delta A)$$
$$\overline{B} = (\Delta A)^{-1}(\exp(\Delta A) - I) \cdot \Delta B$$

По сути получили RNN. Но его можно вычислять с помощью свертки на GPU.

Selective State Space Model

Увеличиваем число параметров. Добавим зависимость матриц **B, C и шага delta от x** (на картинке выделено **красным**). Авторы используют 1 линейный слой в качестве такой функции:

$$s_B(x) = \text{Linear}_N(x),\ s_C(x) = \text{Linear}_N(x),\ s_\Delta(x) = \text{Broadcast}_D(\text{Linear}_1(x)),\ \text{and}\ \tau_\Delta = \text{softplus}$$

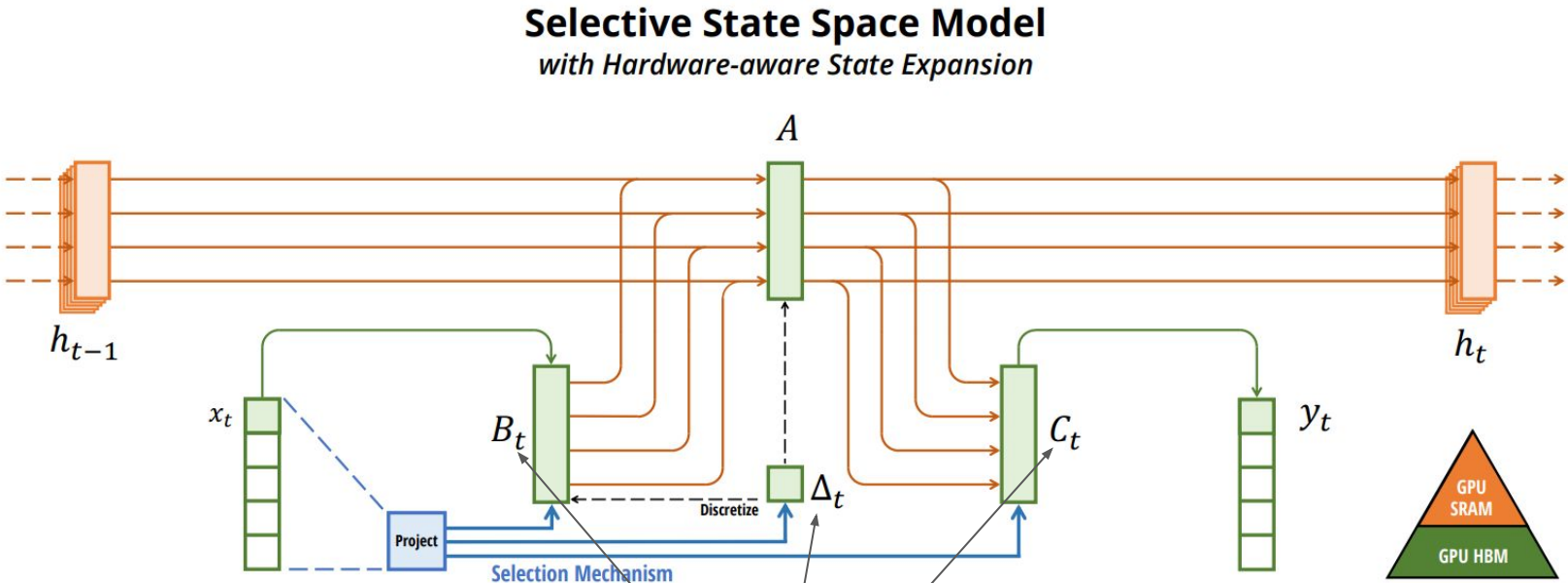
Algorithm 1 SSM (S4)

Input: $x : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
Output: $y : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
1: $\boldsymbol{A} : (\mathtt{D}, \mathtt{N}) \leftarrow$ Parameter
▷ Represents structured $N \times N$ matrix
2: $\boldsymbol{B} : (\mathtt{D}, \mathtt{N}) \leftarrow$ Parameter
3: $\boldsymbol{C} : (\mathtt{D}, \mathtt{N}) \leftarrow$ Parameter
4: $\Delta : (\mathtt{D}) \leftarrow \tau_\Delta(\text{Parameter})$
5: $\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathtt{D}, \mathtt{N}) \leftarrow \text{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
6: $y \leftarrow \text{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-invariant: recurrence or convolution
7: **return** y

Algorithm 2 SSM + Selection (S6)

Input: $x : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
Output: $y : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
1: $\boldsymbol{A} : (\mathtt{D}, \mathtt{N}) \leftarrow$ Parameter
▷ Represents structured $N \times N$ matrix
2: $\boldsymbol{B} : (\mathtt{B}, \mathtt{L}, \mathtt{N}) \leftarrow s_B(x)$
3: $\boldsymbol{C} : (\mathtt{B}, \mathtt{L}, \mathtt{N}) \leftarrow s_C(x)$
4: $\Delta : (\mathtt{B}, \mathtt{L}, \mathtt{D}) \leftarrow \tau_\Delta(\text{Parameter} + s_\Delta(x))$
5: $\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathtt{B}, \mathtt{L}, \mathtt{D}, \mathtt{N}) \leftarrow \text{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
6: $y \leftarrow \text{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-varying: recurrence (*scan*) only
7: **return** y

Selective State Space Model



Зависимость параметров от входа (Selection Mechanism)

Параметры SSM

Selective State Space Model

Процесс, заданный диффуром:

$$h'(t) = Ah(t) + Bx(t)$$
$$y(t) = Ch(t)$$

x(t) — вход
h(t) — скрытое состояние
y(t) — выход

Дискретизация диффура с шагом delta: $(\Delta, A, B, C) \mapsto (\overline{A}, \overline{B}, C)$

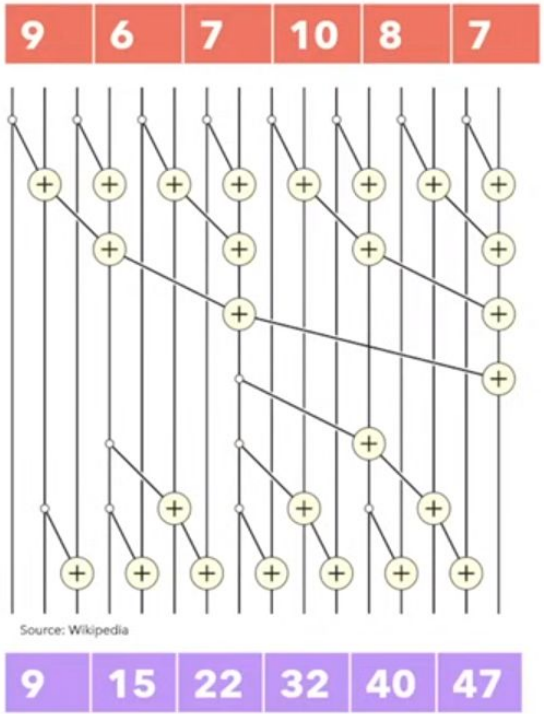
B, C, delta — параметры, **зависящие от x**

$$h_t = \overline{A}h_{t-1} + \overline{B}x_t$$
$$y_t = Ch_t$$

$$\overline{A} = \exp(\Delta A)$$
$$\overline{B} = (\Delta A)^{-1}(\exp(\Delta A) - I) \cdot \Delta B$$

Selective State Space Model

Теперь на каждом объекте матрицы A, B, C разные, поэтому нельзя использовать свертку. Авторы придумали другой алгоритм, который назвали **scan**.



$$y_k = C\bar{A}^k\bar{B}x_0 + C\bar{A}^{k-1}\bar{B}x_1 + \cdots + C\bar{A}\bar{B}x_{k-1} + C\bar{B}x_k$$

Ассоциативность операции умножения:
(X * Y) * Z = X * (Y * Z)

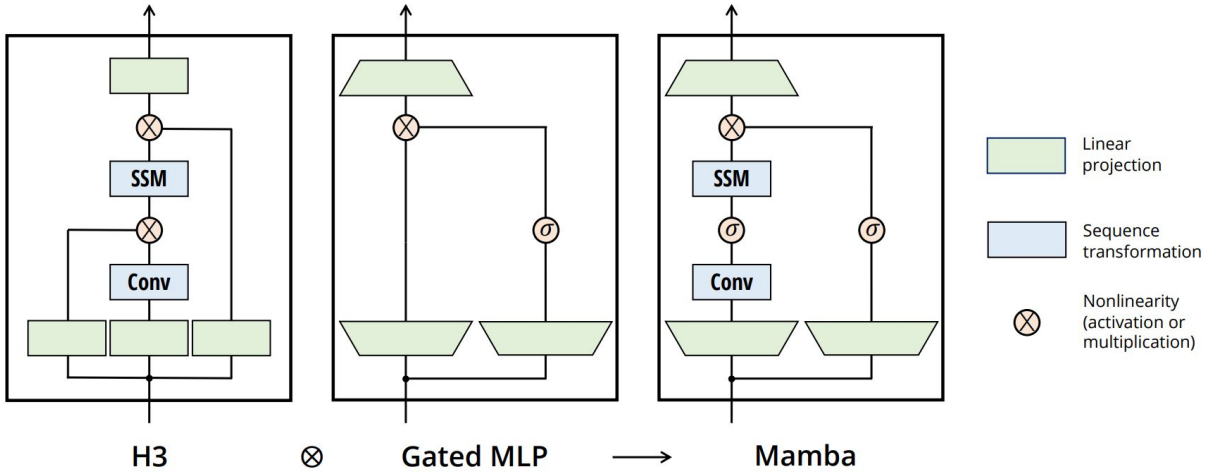
O(N) -> O(N / (число тредов))

Mamba

Mamba (3 картинка) – это комбинация SSM и сверток

В Mamba вход проходит в следующем порядке через слои:

- Линейные слои (зеленые).
- **Свертка** (Conv) и **SSM**.
- **Skip-connections**
- Выход SSM и **skip-connections** перемножаются



Авторы скомбинировали 2 существующие модели (H3 и Gated MLP) и получили Mamba

Эксперименты

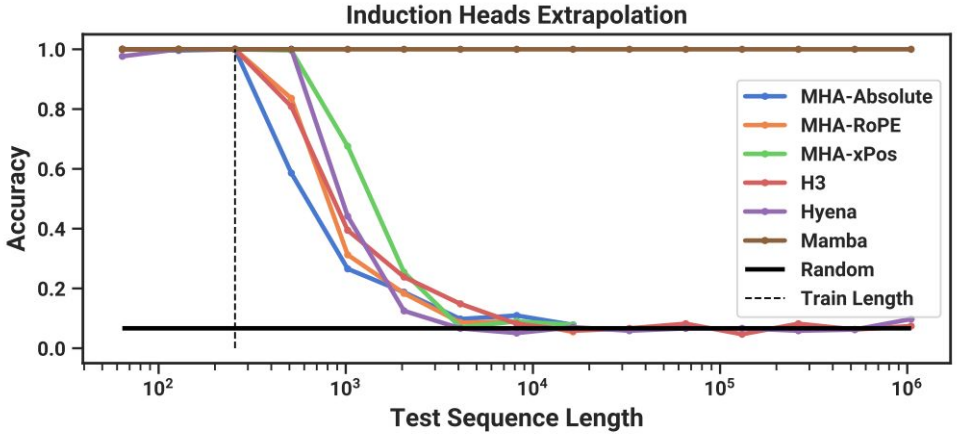
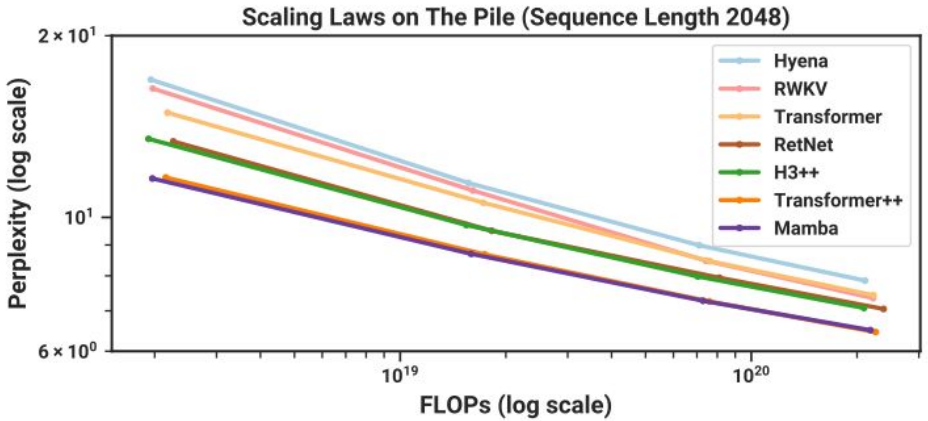


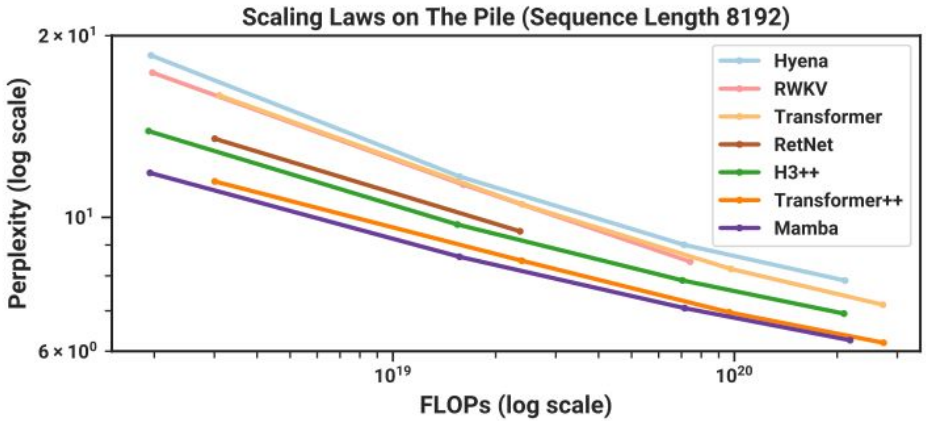
Table 2: (**Induction Heads.**) Models are trained on sequence length $2^8 = 256$, and tested on increasing sequence lengths of $2^6 = 64$ up to $2^{20} = 1048576$. Full numbers in Table 11.

Задача: Если модель ранее видела биграмму в последовательности, например, “Harry Potter”, то в следующий раз, когда “Harry” появится в этой же последовательности, модель должна продолжить “Potter”

С **ростом длины входа качество Mamba не падает**.
Mamba может работать с очень длинными текстами без потери качества и скорости.

Число операций vs Качество

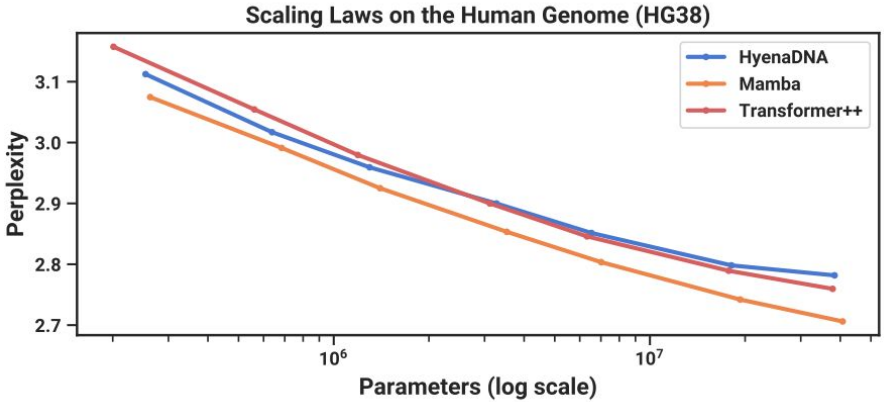


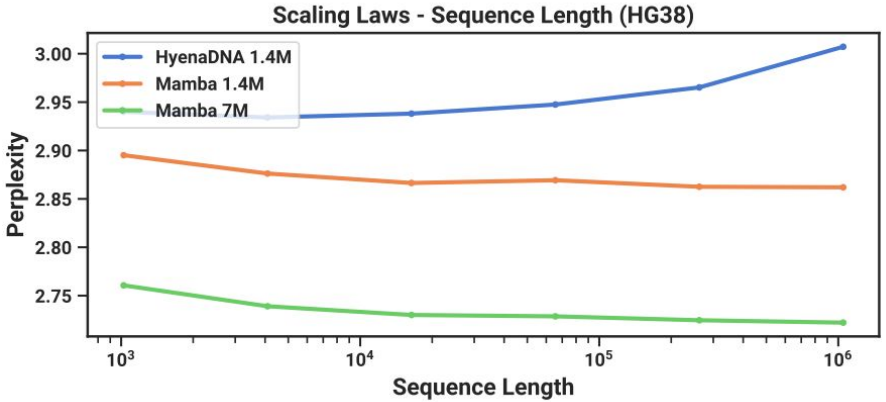


Задача предсказания следующего слова.

- Эксперимент сравнивает **качество одинаково сложных** моделей
- Mamba при том же числе операций FLOPs-ов добивается более хорошего качества, чем другие архитектуры.
- Качество Transformer++ и Mamba **почти одинаковое**

DNA





Задача: предсказание следующего элемента в последовательности генома человека. DNA – это **длинные последовательности**, для них Mamba справляется лучше трансформеров.

Практическое применение Mamba – **работа с последовательностями, которые длиннее 10 000 слов**. Здесь трансформеры начинают уступать. Для всех остальных задач стоит использовать трансформеры.

Плюсы и минусы Mamba

Плюсы:

- Оптимизирована для более **быстрого вычисления на GPU**
- Эффективно на **длинных последовательностях: O(N)**
- Первая **модель без attention**, которая приближается к трансформерам

Минусы:

- Авторы не экспериментировали с очень **большими моделями** трансформеров
- На задачах с небольшими последовательностями Mamba **не показывает прироста в качестве**

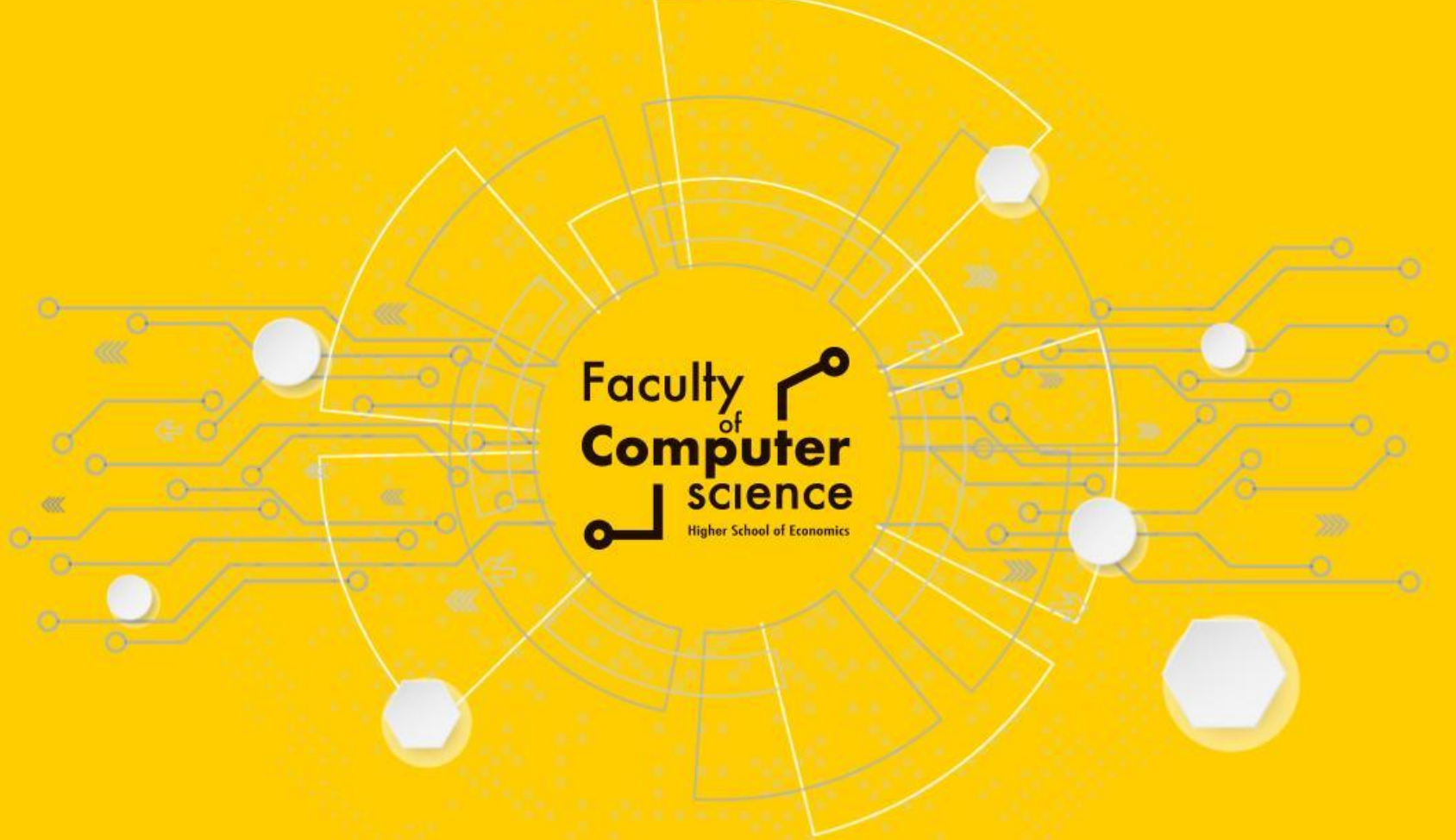
Практический вывод

Последовательность длиннее 10 000 токенов?

- Да => Mamba
- Нет => Transformer

Ссылки

- Оригинальная статья (сложна для понимания): https://arxiv.org/ftp/arxiv/papers/2312/2312.00752.pdf
- Гайд по Mamba: https://www.youtube.com/watch?v=8Q_tqwpTpVU
- Более простой гайд по Mamba: https://www.youtube.com/watch?v=9dSkvxS2EB0



mskazadaev@edu.hse.ru

SSM (State Space Model)

По сути получили RNN. Но его можно вычислять с помощью свертки на GPU.

$$h_t = \overline{A}h_{t-1} + \overline{B}x_t$$
$$y_t = Ch_t$$

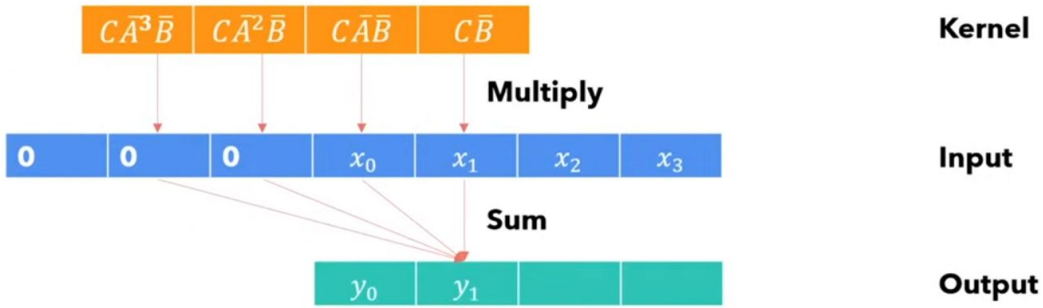
$\longrightarrow$

$$y_k = C\bar{A}^k\bar{B}x_0 + C\bar{A}^{k-1}\bar{B}x_1 + \cdots + C\bar{A}\bar{B}x_{k-1} + C\bar{B}x_k$$

Одинаковые матрицы для каждого объекта в выборке, поэтому их считаем один раз для батча

$$\overline{\boldsymbol{K}} = (C\overline{\boldsymbol{B}}, C\overline{\boldsymbol{AB}}, \dots, C\overline{\boldsymbol{A}}^{k}\overline{\boldsymbol{B}}, \dots)$$

$$y = x * \overline{\boldsymbol{K}}$$



Mamba

Интерпретация архитектуры:

1. **Свертка.** Нужна для того, чтобы перемешать входы. **Решает проблему RNN (слабый эффект далеких входов)**, т. к. теперь каждый input будет состоять не только из n-го входа, но из остальных. Каждый вход может поучаствовать на каждом шаге RNN.
2. **Selective SSM** – по сути улучшенная RNN для учета временных зависимостей. В RNN иногда при увеличении длины контекста качество наоборот падает. В SSM матрицы зависят от входа, что позволяет модели эффективно фильтровать шум, обнуляя похожие на шум входы, например, устремляя delta к infinity. Именно поэтому модель называется **Selective**.