

Victor Nosko, Zachar Ponimash



Avatar Machine LLC

# Fast experts tuning: a better domain adaptation method for transformers efficient tuning

**Interpretable Natural Language Processing (INLP)**  
AGI-22 Workshop | 19-22 August, 2022

# Main idea

## Challenge

Generative models require huge GPU power to train all weights

## Solution

Train not the entire neural network, but only one or several layers

## Outline of the report:

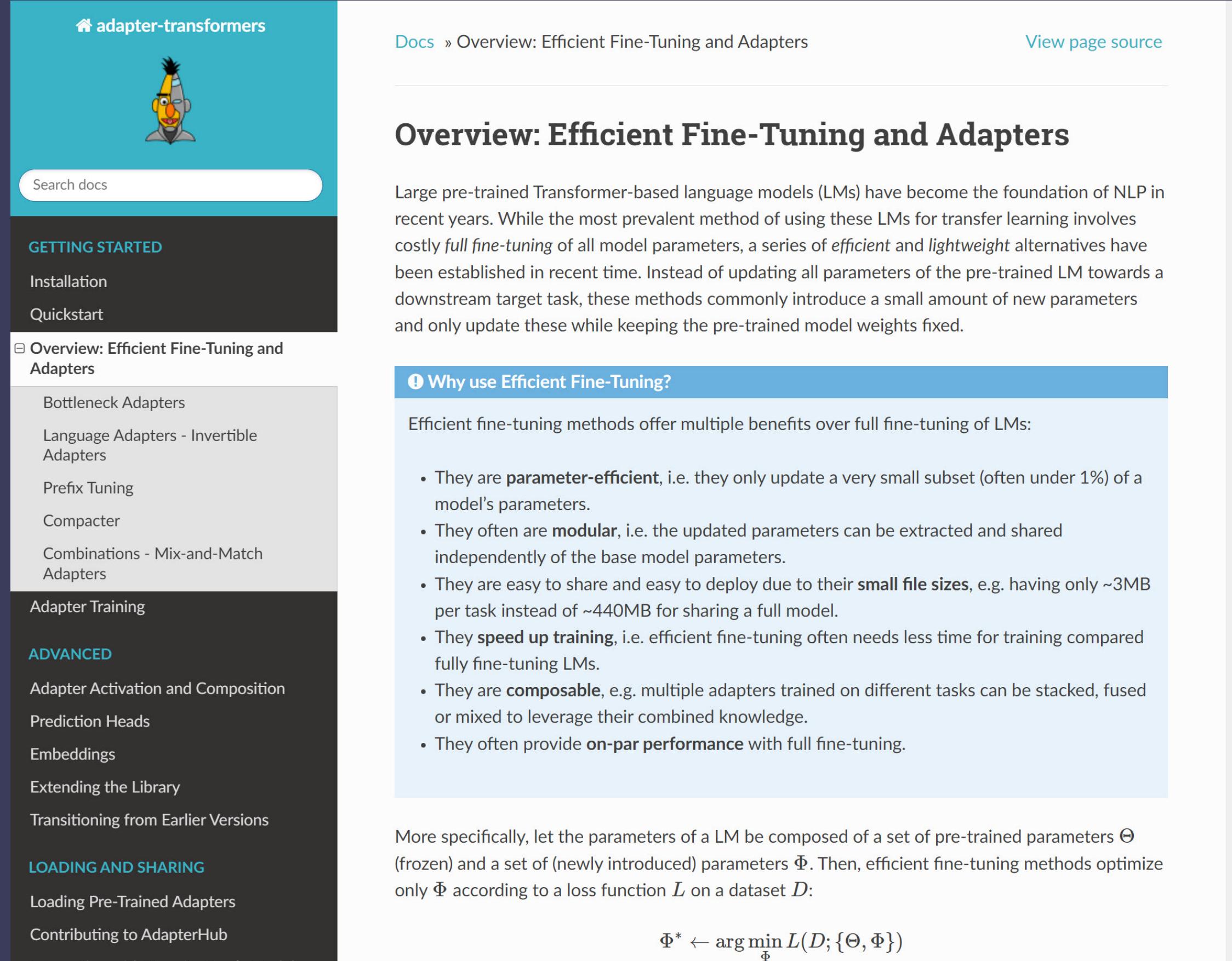
- 1) About adapters
- 2) Types of adapters
- 3) MoE from Google
- 4) **Fast Experts Tuning** technology, the idea is similar to Google's Mixture of Experts.
- 5) GPTFastTuning library: code and Github
- 6) Interpretable generation with BART
- 7) Interpretable generation with ruGPT3

For the first time!



# About adapters

## Adapter-transformers



The screenshot shows the Adapter-transformers documentation site. The top navigation bar includes a logo of a cartoon character, a search bar, and links for 'Docs' and 'View page source'. The main content area has a title 'Overview: Efficient Fine-Tuning and Adapters'. Below it, a section titled 'Why use Efficient Fine-Tuning?' lists several benefits. At the bottom, there's a mathematical formula for efficient fine-tuning:  $\Phi^* \leftarrow \arg \min_{\Phi} L(D; \{\Theta, \Phi\})$ .

- Docs » Overview: Efficient Fine-Tuning and Adapters
- View page source
- Overview: Efficient Fine-Tuning and Adapters**
- Large pre-trained Transformer-based language models (LMs) have become the foundation of NLP in recent years. While the most prevalent method of using these LMs for transfer learning involves costly *full fine-tuning* of all model parameters, a series of *efficient* and *lightweight* alternatives have been established in recent time. Instead of updating all parameters of the pre-trained LM towards a downstream target task, these methods commonly introduce a small amount of new parameters and only update these while keeping the pre-trained model weights fixed.
- Why use Efficient Fine-Tuning?**
- Efficient fine-tuning methods offer multiple benefits over full fine-tuning of LMs:
  - They are **parameter-efficient**, i.e. they only update a very small subset (often under 1%) of a model's parameters.
  - They often are **modular**, i.e. the updated parameters can be extracted and shared independently of the base model parameters.
  - They are easy to share and easy to deploy due to their **small file sizes**, e.g. having only ~3MB per task instead of ~440MB for sharing a full model.
  - They **speed up training**, i.e. efficient fine-tuning often needs less time for training compared fully fine-tuning LMs.
  - They are **composable**, e.g. multiple adapters trained on different tasks can be stacked, fused or mixed to leverage their combined knowledge.
  - They often provide **on-par performance** with full fine-tuning.
- More specifically, let the parameters of a LM be composed of a set of pre-trained parameters  $\Theta$  (frozen) and a set of (newly introduced) parameters  $\Phi$ . Then, efficient fine-tuning methods optimize only  $\Phi$  according to a loss function  $L$  on a dataset  $D$ :
$$\Phi^* \leftarrow \arg \min_{\Phi} L(D; \{\Theta, \Phi\})$$

## Adapter types

Efficient fine-tuning methods offer multiple benefits over full fine-tuning of LMs:

They are parameter-efficient, i.e. they only update a very small subset (often under 1%) of a model's parameters.

They often are modular, i.e. the updated parameters can be extracted and shared independently of the base model parameters.

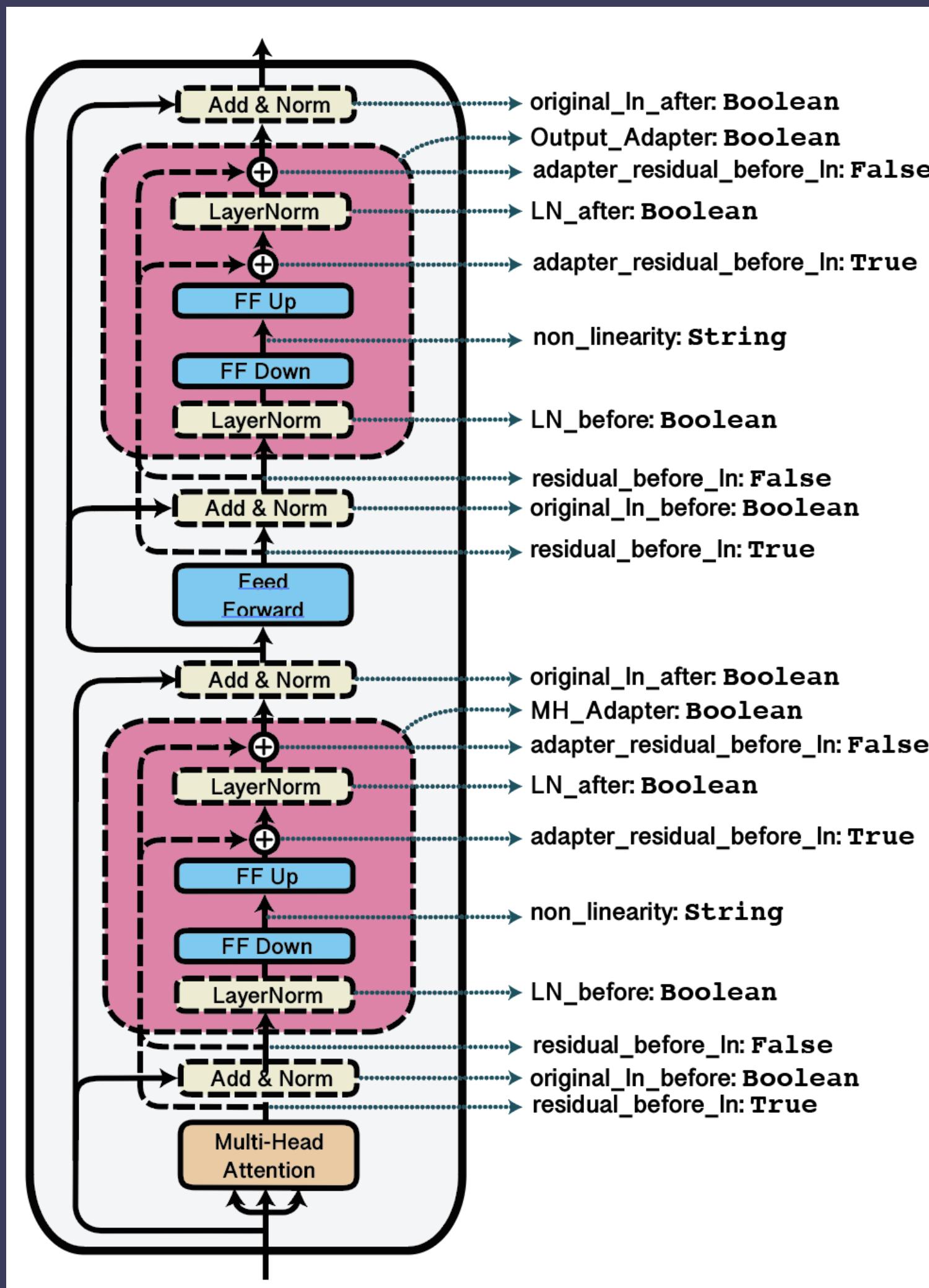
They are easy to share and easy to deploy due to their small file sizes, e.g. having only ~3MB per task instead of ~440MB for sharing a full model. They speed up training, i.e. efficient fine-tuning often needs less time for training compared fully fine-tuning LMs.

They are composable, e.g. multiple adapters trained on different tasks can be stacked, fused or mixed to leverage their combined knowledge.

More: <https://docs.adapterhub.ml/overview.html>

# What is Adapters?

## Bottleneck adapter scheme



## Adapter types

Outline here:

1. Bottleneck
2. AdapterFusion

Others:

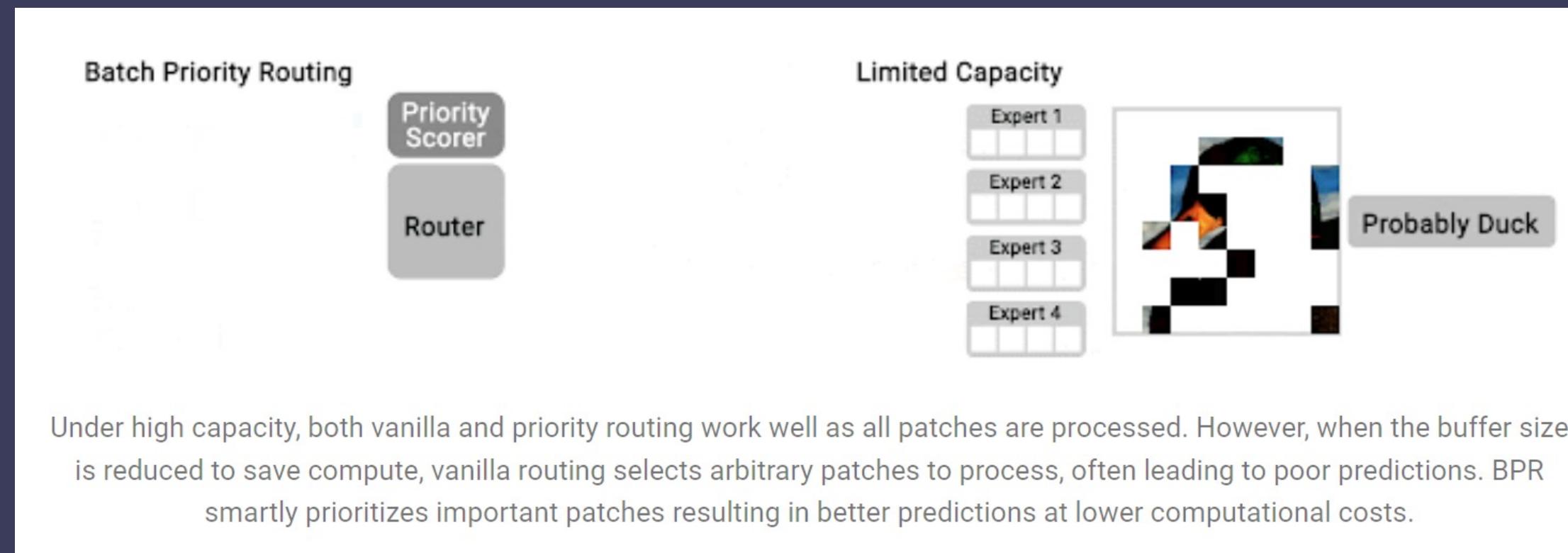
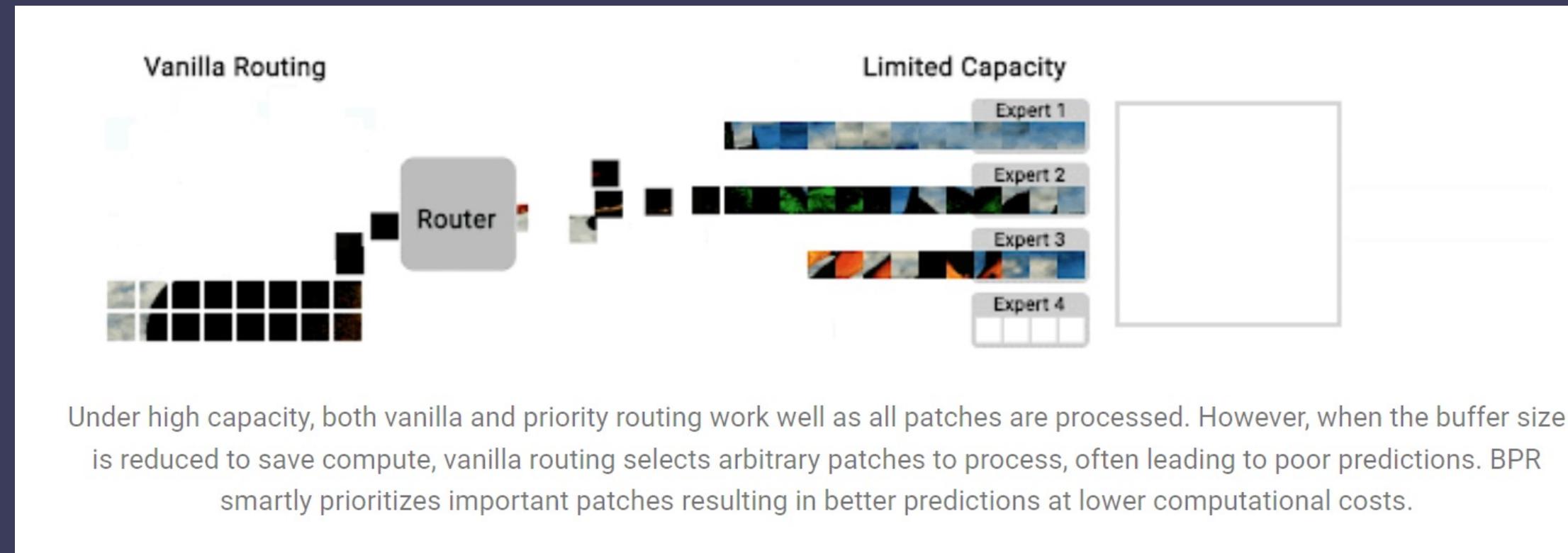
1. MAD-X, Invertible adapters
2. AdapterDrop
3. MAD-X 2.0, Embedding training
4. Prefix Tuning
5. LoRA
6. Parallel adapters,
7. Mix-and-Match adapters
8. Compacter

<https://aclanthology.org/2021.eacl-main.39.pdf>

Подробнее: <https://github.com/adapter-hub/adapter-transformers#implemented-methods>

# Разница FET и MoE

## MoE from Google:



## Mixture of concepts

What is the concept behind:  
Bottleneck + MoE

There is a layer after LayerNorm layer or several layers of experts. At the output of the model, one or more probabilistic expert models are "attached". The probabilistic model allows you to learn faster

More: <https://ai.googleblog.com/2022/01/scaling-vision-with-sparse-mixture-of.html>

# Fast Experts Tuning technology

Training log (Training time on 1 GPU is 85 seconds), 6 epochs:

```
Downloading: 100% [██████████] 1.61G/1.61G [00:50<00:00, 34.2MB/s]
```

```
[ ] 1 creator = SimpleCreator(model, tokenizer)
```

```
1 creator.train([t1, t2, t3, t4, t5, t6, t7, t8, t9])
```

```
Запустить код в ячейке (Ctrl+Enter)  
Код ячейки не выполнялся в этом сеансе
```

```
Выполнил пользователь Avatar Cybertronics  
пятница, 7 января 2022 г.
```

```
Время выполнения: 85.119 сек.
```

```
29/29 [=====] - 2s 21ms/step - loss: 1.4996
```

```
=====] - 0s 16ms/step - loss: 0.8050
```

```
=====] - 0s 16ms/step - loss: 0.6060
```

```
29/29 [=====] - 0s 16ms/step - loss: 0.4014
```

```
Epoch 4/6
```

```
29/29 [=====] - 0s 16ms/step - loss: 0.4253
```

```
Epoch 5/6
```

```
29/29 [=====] - 0s 15ms/step - loss: 0.4253
```

```
Epoch 6/6
```

```
29/29 [=====] - 0s 16ms/step - loss: 0.3771
```

Training log (Training time on 1 CPU is 1 minute 22 seconds), 3 epochs:

```
Epoch 1/3
```

```
57/57 [=====] - 22s 369ms/step - loss: 2.3544
```

```
Epoch 2/3
```

```
57/57 [=====] - 22s 389ms/step - loss: 1.2231
```

```
Epoch 3/3
```

```
57/57 [=====] - 21s 372ms/step - loss: 0.8616
```

# FastExperts Tuning is **50+** times faster than fine-tuning

**Task: Generating an article or replica on a given topic, so as not to train the entire neural network, because it is long and expensive**

- 1) 50-100 times faster than fine-tuning, for different models
- 2) 50 times faster for the ruGPT3 medium model
- 3) How it works: instead of the full number of parameters, we train individual layers with a smaller number
- 4) GPU training is more than 2 times faster than CPU, on average

# Fast Experts Tuning: generation results

1 t1 = 'S.T.A.L.K.E.R. – серия игр, разработанная украинской компанией GSC Game World. Создана в жанре шутера от первого лица с элементами выживания в постапокалиптическом мире. Игра была выпущена в 2007 году.'  
2 t2 = 'Hunt: Showdown – компьютерная игра в жанре шутера от первого лица и survival horror, разработанная и изданная американской компанией Behaviour Interactive. Игра была выпущена в 2015 году.'  
3 t3 = 'Half-Life 2 – компьютерная игра, научно-фантастический шутер от первого лица, сиквел Half-Life, разработанный немецкой компанией Valve Corporation. Игра была выпущена в 2004 году.'  
4 t4 = 'Gothic (серия игр) · Gothic, в российских изданиях «Готика» – серия компьютерных ролевых игр в жанре фэнтези. Игры были разработаны немецкой компанией Piranha Bytes. Серия состояла из пяти частей, последняя из которых вышла в 2007 году.'  
5 t5 = 'Counter-Strike – серия компьютерных игр в жанре командного шутера от первого лица, основанная на движке GoldSrc. Игры были разработаны американской компанией Valve Corporation. Серия состояла из пяти частей, последняя из которых вышла в 2012 году.'  
6 t6 = 'No Man's Sky – компьютерная игра в жанре космический приключенческий боевик с открытым миром и элементами выживания. Игра была выпущена в 2016 году.'  
7 t7 = ' официально названная в СНГ как S.T.A.L.K.E.R.[8], – компьютерная игра в жанре шутера от первого лица с элементами выживания в постапокалиптическом мире. Игра была выпущена в 2007 году.'  
8 t8 = '«Тень Чернобыля» создавалась как игра с открытым миром; разработчики намеревались поместить игрока в живущую среду с реальными людьми.'  
9 t9 = 'Игра является сюжетным продолжением S.T.A.L.K.E.R.: Тень Чернобыля. В августе 2012 года, после того, как «Выжить»

## ▼ Генерация

```
1 generator = TextGenerator('new_model')  
[] Special tokens have been added in the vocabulary, make sure the associated word embedding  
01/07/2022 09:01:05 - INFO - happytransformer.happy_transformer - Using model: cuda
```

```
[ ] 1 generator.set_variety_of_answers(0.1)
```

```
[ ] 1 #start_text = 'Вглубь Зоны идут '  
2 start_text = 'Серия '  
3 gen_text = start_text+generator.generate(start_text)
```

Setting `pad\_token\_id` to `eos\_token\_id`:50256 for open-end generation.

```
1 print(textwrap.fill(gen_text))
```

Серия «Сталкер» была выпущена издательством «Эксмо» в 2007 году. Действие фильма происходит в окрестностях Чернобыльской АЭС в зоне отчуждения АЭС «Фукусима». Действие разворачивается вокруг группы военных учёных, пытающихся выяснить причину

# Good controllable generation

# Global task: Generation of continuation text, dialogue

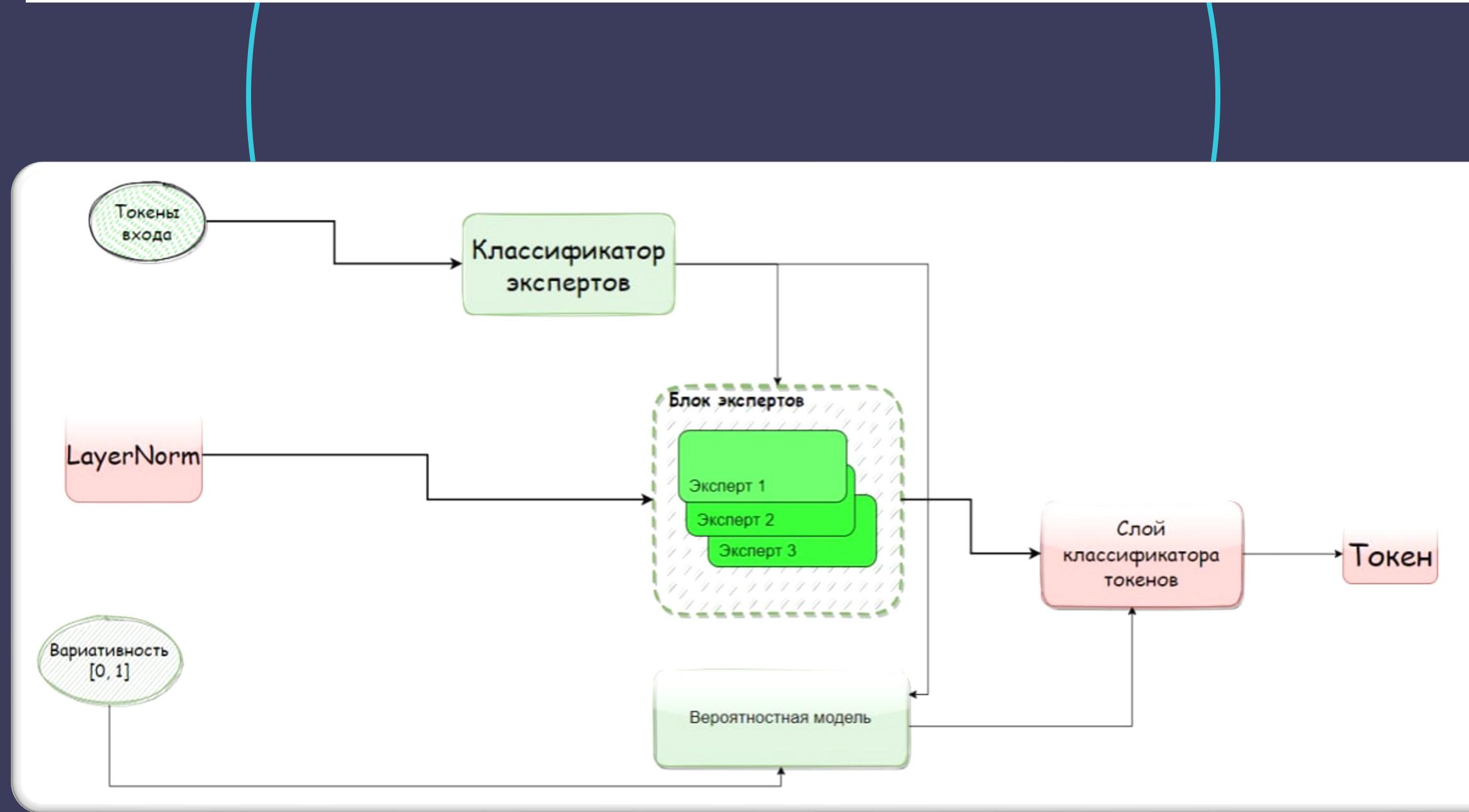
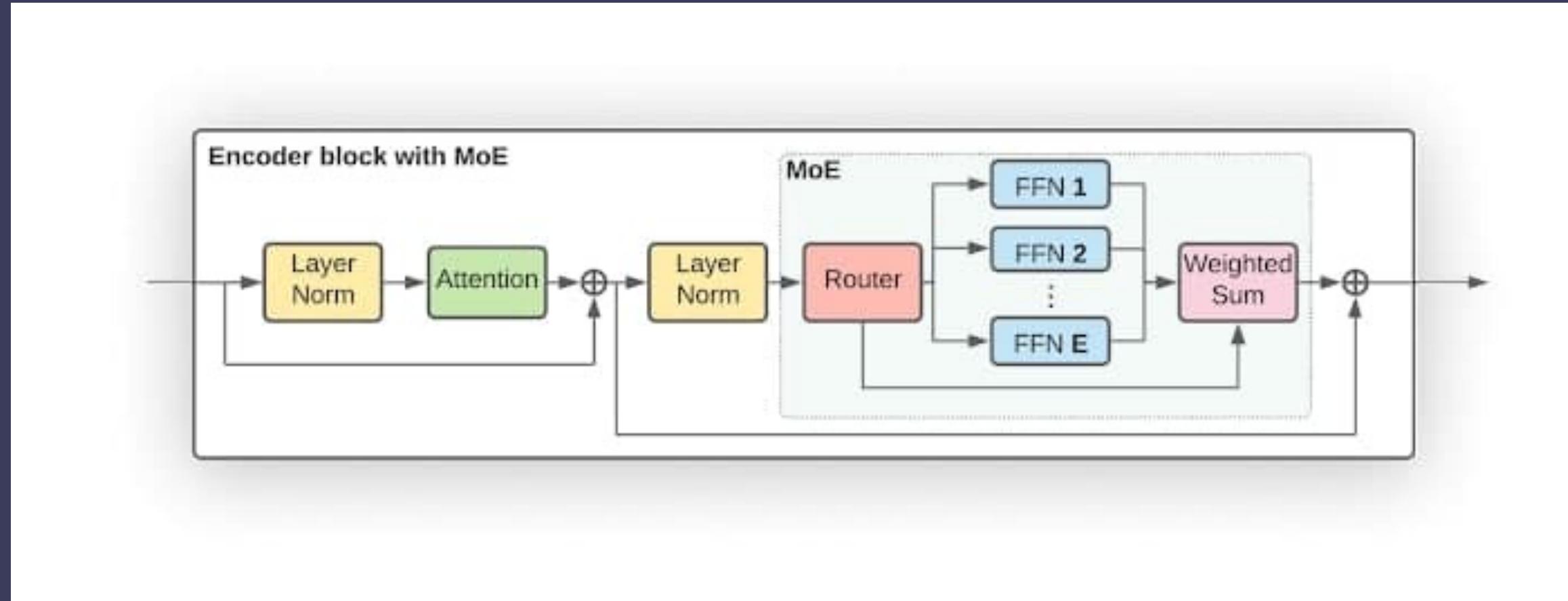
**There is a “set\_variety\_of\_answers” parameter with a value from 0 to 1 that controls the variability. If it is equal to 0, then we follow the dataset, if 1 – almost the original model is used.**

Before training (generation from the word "Stalker"):  
**Сталкер" - это сборник рассказов писателя Сергея Лукьяненко "Сталкер".**

After training (generation from the word "Stalker")  
**Сталкер – игра с открытым миром**

**Result: we got controllable generation**

# Fast Experts Tuning vs MoE

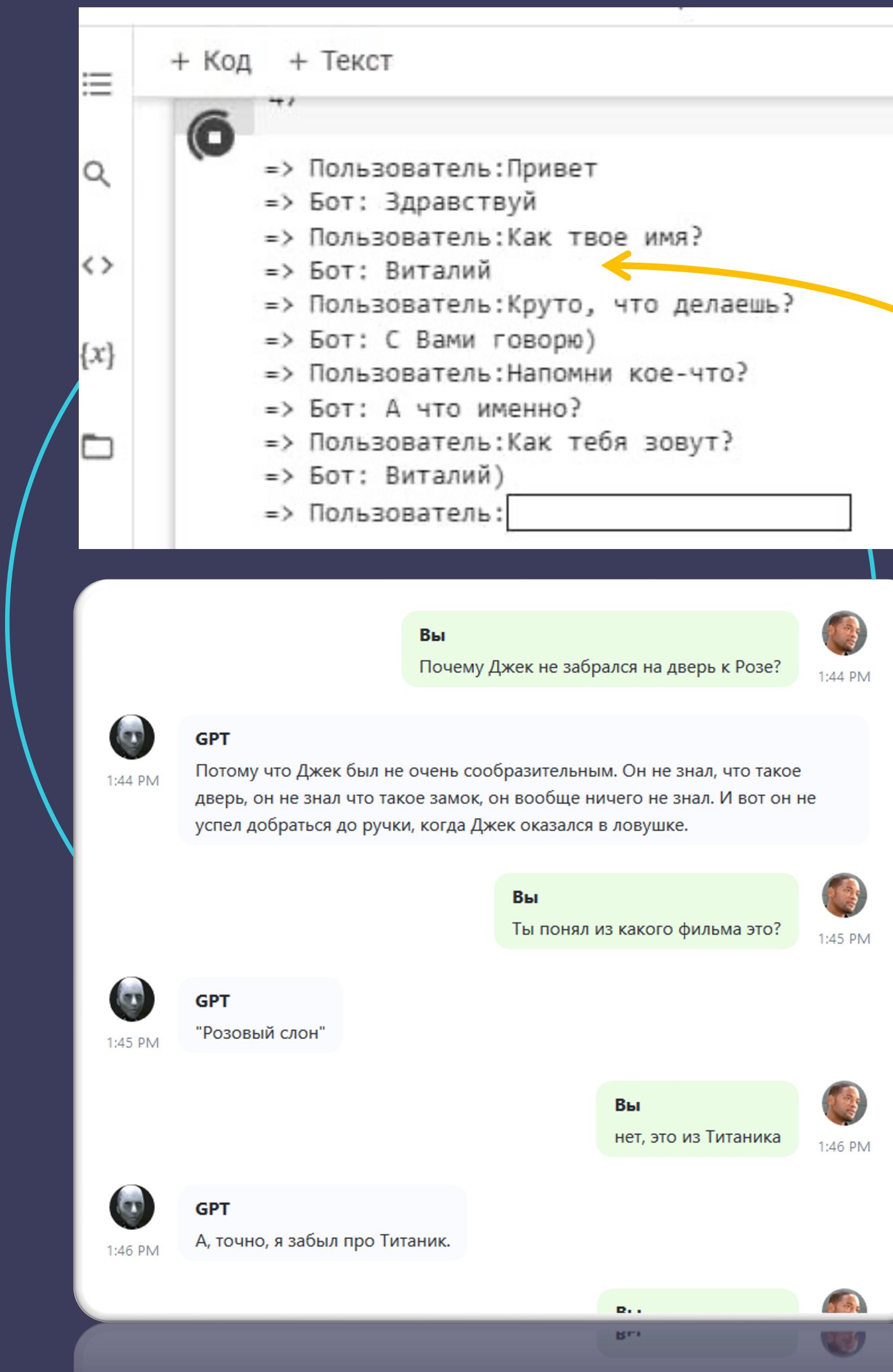


## FET vs MOE

**Task: To speed up controllable generation, but leave the quality of the generated text at the same level**

- 1) An additional model that helps Experts
- 2) There is no Weighted Sum block, but there is an effect of it
- 3) LayerNorm – from GPT
- 4) The idea is similar to Google's MoE, but the implementation is more efficient

# Fast Experts Tuning and other methods



## Effect of FET

**Local task:**  
**Generation control. Obscene expressions disappear, he begins to understand his name.**  
**Fast, useful for product hypotheses**

**Model: rudialogpt3\_medium\_based\_on\_gpt2**

- 1) Generation of coherent, coherent text
- 2) Neural networks trained by this method do not adapt well to new global tasks, for example, it is possible to make a dialog model from a summarizer, but the quality will be worse. (Work in progress: we know how to solve this problem)
- 3) However, this is often not required, because as a rule it is required to train a neural network already for the final task

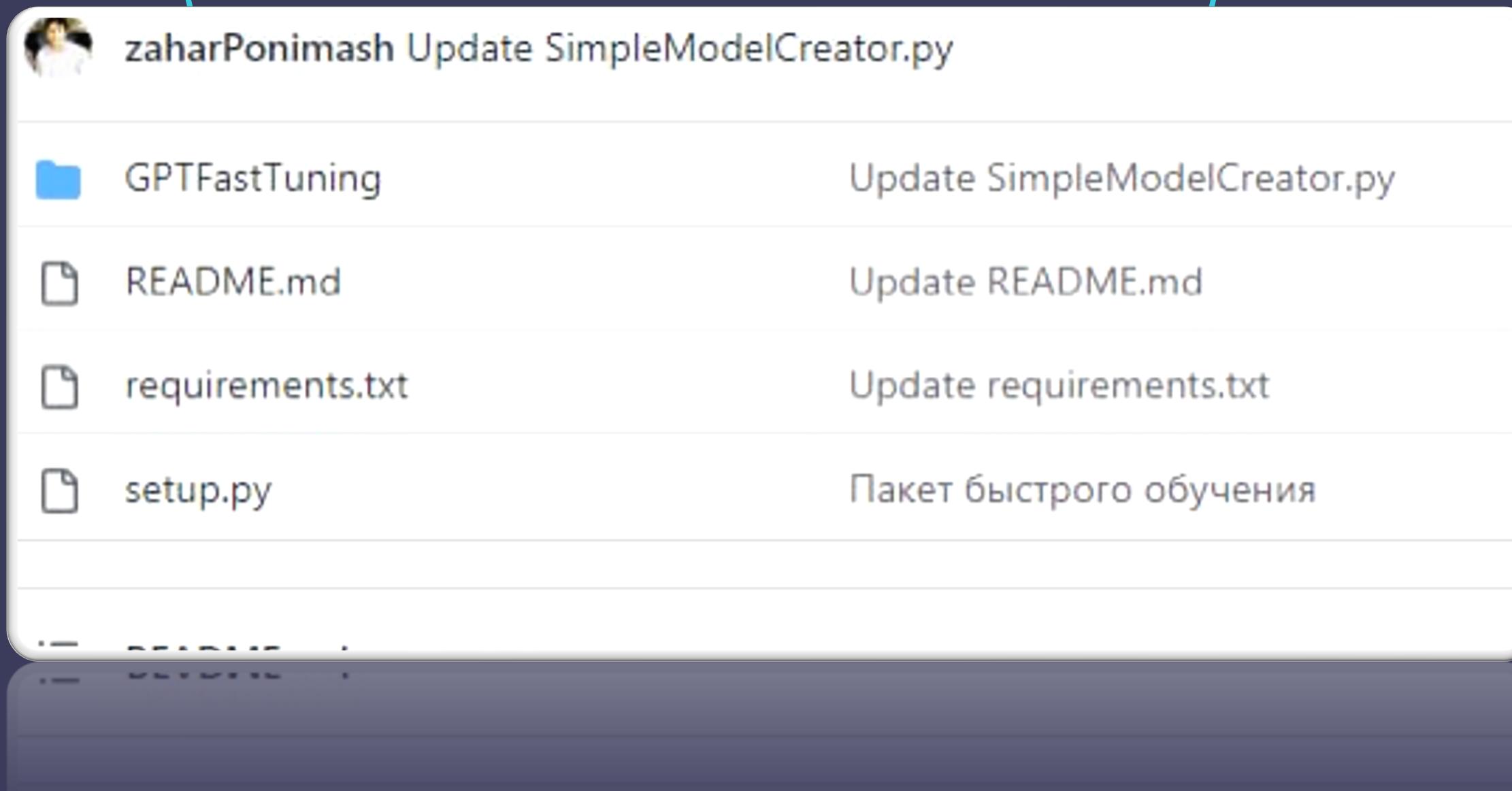
# GPTFastTuning Library

# GPTFastTuning

The library implements the FET learning algorithm

1. Github available (private)
2. Demo on Google Colab
3. Modular and efficient

```
def train(self, data, lr=0.0003, bs = 64, epochs = 6, val_split = 0.0, save_path = 'new_model', d  
x, y = dataset_creator(data, self.cut_embedder, self.tokenizer)  
net = self.trainer.creat_net()  
self.trainer.train(net, x, y, lr=lr, bs = bs, epochs = epochs, val_split = val_split)  
y_set = list(set(y))  
GptBuild(self.trainer, self.gpt_emb, self.tokenizer, y_set, save_path)  
return net
```



# Interpretable generation with BART

```
[12] 1 kw = 'удачи на, желаю тебе'
2 ent = 'тренировка, в 6.00'

1 generate_text(model_bart, tokenizer_bart, kw, ent)
' Удачи на тренировках! Желаю тебе удачи на тренировке в 6.00! '

In [11]: text = 'Несколько недель назад DeepMind пересмотрел выводы Каплана и понял, что, вопреки распространённому
<...>
In [12]: #Генерация
cl = random.choice(data[1].split(','))
gen_bart, gen_gpt = generate_news(k_words=data[0], ent=data[1], emotinons=data[2], cl = cl, longer=146)
gen = textwrap.fill(gen_bart+gen_gpt)
print(gen+'\n\n')

Алексей Сергеевич Каплан: «После нескольких недель назад мы сделали
вывод о том, что у DeepMind есть потенциал для увеличения
вычислительных бюджетов. Исходя из этих выводов, пришли к выводу, что
мы не имеем достаточного ресурса для масштабирования данных». «Мы не
можем позволить себе тратить деньги на то, чего у нас нет. Мы не
сможем обеспечить себя всем необходимым, если не будем иметь
достаточный ресурс для того, чтобы обеспечить потребности наших
клиентов. Поэтому мы приняли решение сконцентрироваться на развитии и
расширении наших возможностей. В этом году мы продолжим расширять наши
возможности, и в следующем году, возможно, мы сможем увеличить наши
вычислительные бюджеты на 20-30%», - сказал Капланд. По словам
Капланда, в будущем году компания планирует увеличить свои возможности
на 30%. «В следующем мы планируем удвоить наши мощности, а в 2022 году
- удвоим их. Это позволит нам увеличить производительность наших
серверов и повысить надежность наших систем.
```

## BART control

---

The library implements the FET learning algorithm

BART generation use entities (common NERs) and make coherent text from them

The quality of generated text is better than GPT

# Interpretable generation with GPT

# AI Генератор статей

Нейросеть напишет любую статью за человека. В этом демо представлена Базовая модель (темы: Экономика, Реклама, Техника, Стиль, Квартира, Искусство) и модель, создающая описания к Детским товарам ecommerce.

[Купить](#) полную версию: лучше качество и уникальность текста, настраиваемая длина статьи, больше ключевых слов, любая ваша тематика и стиль. [Задать вопрос](#)

## ❖ Разработка компании AvatarMachine

Выберите модель нейросети:

Детские товары

Введите ключевые слова (от 1 и до 5):

лего X

игра X

купить X

Нажмите Enter, чтобы добавить еще

Введите текст начала статьи

Лучшие конструкторы для детей

29/200

Количество слов:

100  200 300

❖ Генерация статьи может занять до 5 секунд ❖

Создать текст!

Начинаем генерацию...

в работе 30

...  Готово!

Лучшие конструкторы для детей LEGO в интернет-магазине «Детский мир» Купить конструктор LEGO – значит подарить ребенку увлекательную игру, способствующую развитию логического мышления и воображения. Ассортимент наборов постоянно пополняется актуальными новинками известных брендов, что позволяет подобрать оптимальный вариант как мальчикам, так и девочкам возрастом от 3 до 10 лет. Разнообразие сюжетно-ролевых сценариев на любой вкус позволит найти именно то изделие, которое придется по душе Вашей малышке! Например, серия Lego Super Heroes посвящена популярному мультсериалу о гонщиках Формулы-1. С ее помощью ребенок сможет воссоздать неповторимые эпизоды из любимых серий или придумать собственные захватывающие сюжеты с привлечением к игровому процессу друзей и членов семьи. Или же купить конструктор LEGO Minecraft можно будет уже по мотивам одноименного же клинта конструкторов LEGO Minecraft можно будет уже по мотивам одноименного звездного сюжета с интереснейшим набором изображений на тему Minecraft. Nun можешь воссоздать интересные эпизоды из вашего мира с помощью собственных изображений, созданных на компьютерах Minecraft. С ее помощью ребенок

# GPTFastTuning

**ruGPT3 generation use keywords and make  
coherent text from them.**

# Contacts



Ai visionary, Newton chatbot developer

[ceo@graphgrail.com](mailto:ceo@graphgrail.com)

<http://graphgrail.com/ru>



6 years of experience in AI. Author of his own Ai Framework

[https://github.com/AIFramework/AI\\_Free](https://github.com/AIFramework/AI_Free)

## Victor Nosko

CEO, Ai researcher

## Zachar Ponimash

CTO, Ai researcher

## Interpretable Natural Language Processing (INLP)

AGI-22 Workshop | 19-22 August, 2022

Sign Up!