# Development and application of visual-language models for robotics

Казачков Даниил
Email: kazachkovdaniil.rb@gmail.com
Telegram: @Dan_i_il
GitHub: RoboticVLA
W&B logs: classification, regression

Abstract

This work is devoted to implementing a model based on a transformer architecture for predicting robot actions. To collect the dataset, the $\texttt{RT}-\texttt{1}-\texttt{X}$ checkpoint was used, logging was performed in `wandb`, and $\texttt{nano}-\texttt{GPT}$ was used as the transformer model.

## Introduction

The first attempts at training robots consisted in collecting a dataset and writing task-specific instructions [1, 2], narrowly specialized for a particular task. This approach did not allow the solution to generalize to other domains. At the same time, large language models (LLMs) demonstrated impressive capabilities in solving a wide range of tasks [3, 4, 5], which led to the idea of leveraging their world knowledge to tackle more complex problems. One of the breakthrough works using a transformer architecture was $\texttt{RT}-\texttt{1}$ [6], trained on real-world trajectories and capable of generalizing to commands it had not seen before. Its limitations are:

- imitation-learning-based training scheme,
- generalization only to instructions that are combinations of those seen during training,
- a relatively primitive set of tasks.

In the next generation, $\texttt{RT}-\texttt{2}$ [7], the authors used a vision-language model (VLM) and pretraining on web data, which increased its generalization ability, and the presence of reasoning allowed it to solve more complex tasks. However, several limitations still remained:

- difficulty in acquiring new skills,
- a larger model leads to higher latency.

The authors of the `Octo` model [8] attempted to address the inability to scale the model to completely new tasks. They trained the model on an even larger number of trajectories, improved action planning, adapted the solution to different environments, and enabled the execution of previously unseen tasks.

## Solution

To get more familiar with existing methods, I tried to adapt the transformer model nanoGPT for generating robot actions.

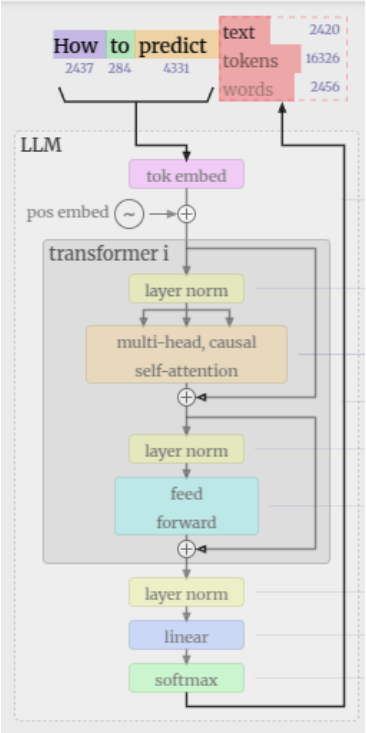


Рис. 1: nanoGPT architecture.

In the $\texttt{RT}-\texttt{1}$ paper [6], the authors discretized actions into 256 bins, and the transformer model solved a classification problem: it predicted the bin index (from 0 to 255) for each component of the action. Logistic regression was used as the loss function. The price for this is an irreducible error equal to $\frac{1}{256} = 0.0039$.

Since I train the `nanoGPT` model from scratch, I decided to evaluate its performance both on a classification task and on a regression task. For the latter, I introduced several important modifications:

- modified the input layer so that it accepts 8-dimensional vectors instead of tokens,
- modified the output layer so that it produces 8-dimensional vectors instead of a probability distribution over tokens,
- used MSE instead of cross-entropy for regression.

### Data collection

The notebook with the code can be found at the following link. The Simpler environment and the pretrained $\texttt{RT}-\texttt{1}-\texttt{X}$ model [9] were used. In total, 134 trajectories were collected

with the structure shown in 1.

Листинг 1: Structure of a trajectory step

```
{
  'timestep': 0,
  'action': {
    'world_vector': array([ 0.1291585 ,  0.10567522, -0.02739727]),
    'rot_axangle': array([ 0.29202676, -0.00307393,  0.18136406]),
    'gripper': array([0.]),
    'terminate_episode': array([0, 1, 0], dtype=int32)
  }
}
```

Then the `terminate_episode` vector is replaced with a single scalar indicating whether to continue or stop the trajectory.

### Training

Training was carried out on an NVIDIA GeForce RTX 2080 Ti GPU. In the project repository, each model has a corresponding `.py` file with its definition and training script.

`nanoGPT` is a lightweight model, so I used 1024 bins for discretization.

### Results

Let us denote the error of the regression and classification models by $L_{\text{reg}} = 0.259$ and $L_{\text{cls}} = 0.267$, respectively. Their values can also be found in the Colab notebook.

## Conclusion

For the regression task, one can use specialized transformer architectures such as `TrajectoryTransformer` or `DecisionTransformer`, each with its own advantages and disadvantages. However, for an initial exploration, a simple model is the most suitable choice. The rather modest performance can be explained by the small size of the dataset, which also caused the classification model to overfit quickly.

## Список литературы


[1] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In Conference on robot learning, pages 651–673. PMLR, 2018.

[2] Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. arXiv preprint arXiv:2104.08212, 2021.

[3] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[4] Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models, 2023.

[5] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[6] Anthony Brohan et al. Rt-1: Robotics transformer for real-world control at scale, 2023.

[7] Anthony Brohan et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023.

[8] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In Proceedings of Robotics: Science and Systems, Delft, Netherlands, 2024.

[9] Embodiment Collaboration et al. Open x-embodiment: Robotic learning datasets and rt-x models, 2024.