Developer Circles Skills Connection

Demystifying the ML Engineer Role
with Olivier Delalleau and Shagun Sodhani

FACEBOOK

Introduction - Olivier

1. Research Engineering Manager at Facebook AI Research

2. Interested in Reinforcement Learning

Background - Olivier

1. MS(-like) in CS
2. Research Assistant at LISA (ex-Mila) for ~4 years
3. PhD in CS/ML
4. AI Programmer (~Data Scientist / ML Engineer) at Ubisoft for 7 years
5. Research Engineering Manager at Facebook AI Research
   (joined 2 years ago)

Introduction - Shagun

1. Research Engineer at Facebook AI Research
2. Interested in Lifelong Learning

Background - Shagun

1. Bachelors in CS
2. ML Engineer at Adobe for 2 years
3. MS in CS
4. Research Engineer at Facebook AI Research (joined 2 years ago)

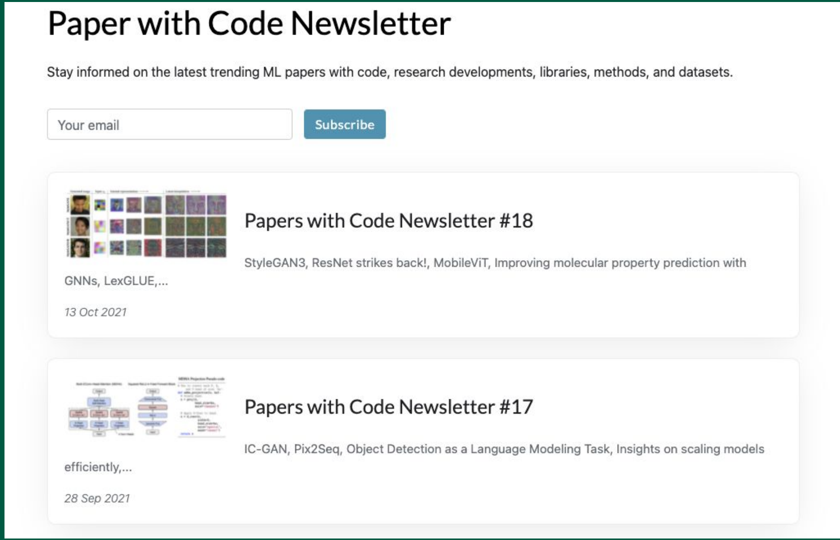
At a high level

1. Mix of research and engineering

2. Three modes of operation (not mutually exclusive)
    - Contributing to open-source
        - Ex: pytorch/fairseq
    - Contributing to fundamental research
        - Ex: Scaling Neural Machine Translation
    - Contributing to products
        - Ex: automatic translation, hate speech and misinformation detection, ads, ...

Day-to-day work

1. Majority of time:
    a. Designing/implementing architectures
    b. Training/debugging models
2. Other time:
    a. People interactions (meetings, presentations, chats, code reviews, planning, …)
    b. Reading/reviewing papers
    c. Keeping up-to date with the tools ecosystem



Skills

1. Excellent programming skills
   (focus on clean and scalable code)

2. Ability to convert latest research papers
   into working algorithms

3. Patience, perseverance to find subtle bugs

4. Learning on the job

5. Communicating effectively

DeepMDP: Learning Continuous Latent Space Models for Representation Learning

6 Jun 2019 · Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, Marc G. Bellemare ·
Edit social preview

Many reinforcement learning (RL) tasks provide the agent with high-dimensional observations that can be simplified into low-dimensional continuous states. To formalize this process, we introduce the concept of a DeepMDP, a parameterized latent space model that is trained via the minimization of two tractable losses: prediction of rewards and prediction of the distribution over next latent states. We show that the optimization of these objectives guarantees (1) the quality of the latent space as a representation of the state space and (2) the quality of the DeepMDP as a model of the environment. We connect these results to prior work in the bisimulation literature, and explore the use of a variety of metrics. Our theoretical findings are substantiated by the experimental result that a trained DeepMDP recovers the latent structure underlying high-dimensional observations on a synthetic environment. Finally, we show that learning a DeepMDP as an auxiliary task in the Atari 2600 domain leads to large performance improvements over model-free RL. (read less)

PDF | Abstract

```python
class Agent(sac_ae.Agent):
    """DeepMDP Agent"""

    def __init__(
        self,
        env_obs_shape: List[int],
        action_shape: List[int],
        action_range: Tuple[int, int],
        device: torch.device,
        actor_cfg: ConfigType,
        critic_cfg: ConfigType,
        decoder_cfg: ConfigType,
        reward_decoder_cfg: ConfigType,
        transition_model_cfg: ConfigType,
        alpha_optimizer_cfg: ConfigType,
        actor_optimizer_cfg: ConfigType,
        critic_optimizer_cfg: ConfigType,
        multitask_cfg: ConfigType,
        decoder_optimizer_cfg: ConfigType,
        encoder_optimizer_cfg: ConfigType,
        reward_decoder_optimizer_cfg: ConfigType,
        transition_model_optimizer_cfg: ConfigType,
        discount: float = 0.99,
        init_temperature: float = 0.01,
        actor_update_freq: int = 2,
        critic_tau: float = 0.005,
        critic_target_update_freq: int = 2,
        encoder_tau: float = 0.005,
        loss_reduction: str = "mean",
        decoder_update_freq: int = 1,
        decoder_latent_lambda: float = 0.0,
        cfg_to_load_model: Optional[ConfigType] = None,
        should_complete_init: bool = True,
    ):
        super().__init__(
```

Tech stack (what we mostly use at FAIR)

1. Language: Python
2. Framework: PyTorch, Hydra
3. Data analysis: Pandas, Jupyter Notebooks
4. Editor: Visual Studio Code
5. Notes/book-keeping: Overleaf, G Docs, Notability

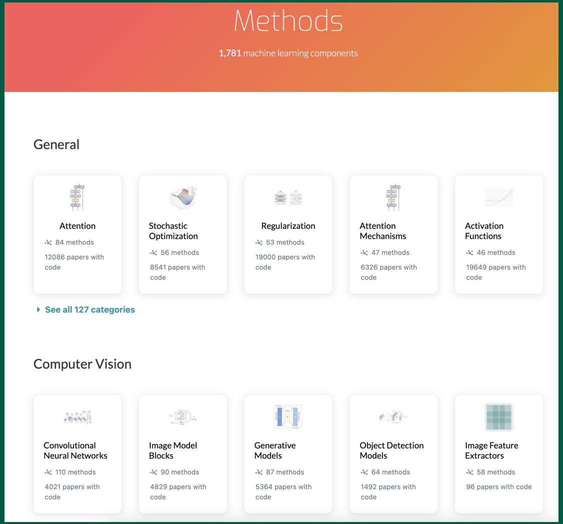
From Research to Production

1. Ensure commitment from both parties
    - Avoid the “do what you can by yourself, we’ll use it if it works” scenario
    - Find a “Champion” in Production
2. Agree on a clear timeline and objective
    - What do we want to achieve and when?
    - How do we measure impact?
3. Validate the data collection strategy early
    - Do we have all the data we need?
    - Are there access restrictions / privacy concerns?
    - Is the data clean?
    - How much data?
4. Think long term
    - Model re-training
    - Monitoring
    - Maintenance

How to get started?

1. Focus first on acquiring the engineering skills, e.g.:
    - The Hitchhiker's Guide to Python has good (though a bit outdated) advice on Python
    - Learn from others by contributing to an OSS project (ex: lots to choose from in https://github.com/facebookresearch)
    - Read/modify existing code to understand how complex algorithms work, e.g., how to scale distributed applications with fairscale

2. Then expand on your research/ML skills
    - Build a solid foundation on at least one research sub-domain (ex: online courses)
    - Read related papers (ex: PWC's Methods Corpus)
    - Tweak an algorithm's existing implementation, run hyper-parameter sweeps and analyze results (to gain a deeper understanding of its behavior)



Career Opportunities

1. Use LinkedIn, friends, work-network to find relevant opportunities
   - For example, attend conferences, give talks (even 5 minute flash talks are a great start).

2. Try getting a referral

3. Different permutations of “applied”, “ML”, “data”, “research”, “researcher”, “engineer”, “developer”, “software”

4. When choosing between options, focus on the team (and actual work) and not the fancy title
   - In particular the balance between “research” and “engineering” may vary a lot

5. Ideally start job-hunting in Nov/Dec (as soon as next year’s positions are up)

Preparing for interviews

- Coding interview: practice through LeetCode / HackerRank
    - Problem solving is only one aspect: also practice clean & efficient coding, verification (running code "by hand" through test cases, including edge cases) and communication (explaining your thoughts to someone else both before and during coding)
- Systems design interview: see e.g. Preparing for the Systems Design and Coding Interview
- Behavioral interview: prepare answers to typical questions (and be ready to provide details!)
- ML interview: make sure you know both the basics and the latest "trendy" topics (Transformers anyone?)

Career Growth as an Engineer

1. Junior Engineer
    - Ability to plan and execute on tasks taking days / weeks
    - Relying on others for direction
    - Core technical work == mostly solo

2. Senior Engineer
    - Ability to plan and execute on projects taking months / years
    - Setting direction for others
    - Core technical work == mix of solo + overseeing others’ work

3. Star Engineer
    - Do whatever you want, everyone adores you

From Engineer to Manager

1. Don’t think of it as a promotion
2. “Full time” vs “Part time” manager roles
3. Be humble and willing to learn
4. Getting ready: mentorships, internships, project management

Things that worked for us

1. Develop the habit of reading technical papers/blogs
    - And write down short notes summarizing key learnings, because you **will** forget

2. Track your time
    - And periodically re-assess whether you are spending it wisely

3. Automate the boring/repetitive stuff
    - It will pay off over time

4. Be a team player
    - Few people can be very successful on their own

Books | Programming

1. The Pragmatic Programmer: your journey to mastery

2. Programming Pearls

3. The Mythical Man Month

Books | Writing

1. The Elements of Style
2. On Writing Well

Books | Machine Learning

1. Pattern Recognition and Machine Learning

2. Deep Learning Book

3. Dive into Deep Learning

Books | Self Improvement

1. Atomic Habits

Books | Broaden Your Perspective

1. Where Good Ideas Come From

2. The Alchemist

3. Thinking Fast and Slow

4. Zero to One: Notes on Startups, or How to Build the Future

Surprises

1. Importance of [deep work]
2. Wasting time in repetitive work, that can be automated