English Title

Deutscher Titel
Bachelor-Thesis von John Doe aus Birthplace
Januar 2018

TECHNISCHE
UNIVERSITÄT
DARMSTADT

English Title
Deutscher Titel

Vorgelegte Bachelor-Thesis von John Doe aus Birthplace

1. Gutachten: Prof. Dr. N. N.
2. Gutachten: Prof. Dr. N. N.
3. Gutachten: Prof. Dr. N. N.

Tag der Einreichung:

For Thomas Hesse and Kevin Luck

Erklärung zur Bachelor-Thesis

Hiermit versichere ich, die vorliegende Bachelor-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
In der abgegebenen Thesis stimmen die schriftliche und elektronische Fassung überein.

Darmstadt, den 8. Januar 2018

(John Doe)

Thesis Statement

I herewith formally declare that I have written the submitted thesis independently. I did not use any outside support except for the quoted literature and other sources mentioned in the paper. I clearly marked and separately listed all of the literature and all of the other sources which I employed when producing this academic work, either literally or in content. This thesis has not been handed in or published before in the same or similar form.
In the submitted thesis the written copies and the electronic version are identical in content.

Darmstadt, January 8, 2018

(John Doe)

Abstract

Reinforcement learning relies on policy gradient but the gradient is known only in expectation and most of the time stochastic policies. This leaves some room for zero order methods and BO can combine solving the problem and the exploration strategy from deterministic policies. We investigate in this paper how to integrate efficient exploration strategies stemming from Bayesian optimization for solving high dimensional reinforcement learning problems. We propose a novel optimization algorithm that is able to scale Bayesian optimization to such high dimensional tasks by restricting the search to the local vicinity of a search distribution and by proposing kernels capturing similarity in behavior rather than parameter. We show in the experiments that our approach can be very useful for applications such as robotics.

Zusammenfassung

Das Ziel im bestärkten Lernen ist das Finden einer Strategie, welche die erhaltene Belohnung eines Agenten maximiert. Da der Suchraum für mögliche Strategien sehr groß sein kann, verwenden wir Bayesian optimization, um die Anzahl der Evaluierungen durch den Agenten zu minimieren. Das hat den Vorteil, dass zeit- und kostenaufwändige Abläufe, wie beispielsweise das Bewegen eines Roboterarms, reduziert werden. Die effektivität der Suche wird maßgeblich von der Wahl des Kernels beeinflusst. Standardkernel in der Bayesian optimization vergleichen die Parameter von Strategien um eine Vorhersage über bisher nicht evaluierte Strategien zu treffen.

Der Trajectorykernel vergleicht statt der Parameter, die aus den jeweiligen Strategien resultierenden Verhaltensweisen. Dadurch werden unterschiedliche Strategien mit ähnlichem Resultat von der Suche weniger priorisiert.

Wir zeigen die Überlegenheit des verhaltensbasierten Kernels gegenüber dem parameterbasierten anhand von Robotersteruerungssimulationen.

Acknowledgments

Contents

Figures and Tables

List of Figures

List of Tables

Abbreviations, Symbols and Operators

List of Abbreviations

Notation	Description
i.i.d.	independently and identically distributed

List of Symbols

Notation	Description
$\boldsymbol{\theta}$	vector of parameters from a probability distribution

List of Operators

Notation	Description	Operator
ln	the natural logarithm	$\ln(\bullet)$

1 Introduction

1.1 Getting started

1.1.1 Installing Glossaries

Note for Windows users: While the makeglossaries command is a perl script for Unix users, there is also a .bat version of the file for Windows users. However, I don't know how to set up MIKTex or equivalent to use this package. Feel free to add a comment if you can add information about this step.

1. **Get and unzip the glossaries package.** I downloaded it from here. Though you can download the source and compile, I found it much easier to simply download the tex directory structure (tds) zip file. Unfortunately, the texlive-latex-extra package available on ubuntu or kubuntu does not contain the glossaries package – it only contains glossary and acronym. I unzipped the contents of the zip file into a directory called "texmf" in my home directory. You'll also want to run "texhash /texmf/" to update the latex database, according to the INSTALL instructions.

2. **(Optionally) get the xfor package.** If your system is like mine, after you've installed the glossaries package latex will complain that it doesn't have the xfor package (which also is not available via apt-get in Ubuntu). Download this package from here.

3. **Open the glossaries zip as root in a nautilus window, terminal window, or equivalent.** You'll be copying the contents to various locations in the root directory structure, and will need root access to do this.

4. **Find the location of your root texmf directory.** In Karmic, this is /usr/share/texmf/, though it may be in another location on your system. Generally, you should have a local texmf folder, i.e. /texmf/, when receiving the IAS slide LaTeX template.

5. **Copy the contents of the tex and doc directories from the glossaries zip into the matching directory structure in your texmf directory.** For me, this meant copying the "doc/latex/glossaries" subdirectory in the zip file to "/usr/share/texmf/doc/latex/", and the same for the tex directory (copy "tex/latex/glossaries" subdirectory in the zip file to "/usr/share/texmf/tex/latex/"). In theory, you can also copy the scripts/ directory in the same way, but I did step 6 instead, as this is what was suggested in the INSTALL document.

6. **Update the master latex database.** Simply run the command "sudo mktexlsr"

7. **Add the location of your scripts/glossaries directory to your $PATH.** This gives programs access to makeglossaries, the perl script you will be using (if you're in linux/unix). If you followed my default instructions in step 1, this location will be "/home/yourname/texmf/scripts/glossaries".

8. **Test the installation.** Change into the directory you created in step 1, into the "doc/latex/glossaries/samples/" subdirectory. There, run "latex minimalgls". If you get an error about xfor, please see step 9. Otherwise, run "makeglossaries" and then "latex minimalgls" again. If everything works, the package is set up for command-line use. You may wish to modify your Kile setup to use glossaries – go to step 10 if this is the case.

9. **Set up the xfor package.** Run steps 3-6 again, but with the xfor.tds.zip file instead of the glossaries zip file. This package is simpler than glossaries, and does not contain a scripts/ subdirectory, so you will not need to do step 7. After installation, try running step 8 again: everything should work.

Source: link

1.1.2 Configure the Modification of the TU Design

You can find in the IAS Thesis folder a folder with the name *texmf*. This folder includes some modifications of the TUD design, for example an updated Thesis statement in english and german. After the installation of the TUD design (`http://exp1.fkp.physik.tu-darmstadt.de/tuddesign/`) you have to move the folder to your home folder. If the folder already exists, then move only the tud-files. Then you have to run the command *texhash ∼/texmf* such that Latex can use the new files. Please note, that the texmf folder already includes some adaptions for the tud-beamer template. If you want to use the original TUD design again, rename the texmf folder and run texhash again.

If you have questions regarding the modifications of the TU design or suggestions, please let me know and send me an email to luck@ias.tu-darmstadt.de

1.2 Documentation

1.2.1 Structure of the IAS LaTeX-Framework

The structure of this framework is illustrated in the following figure.

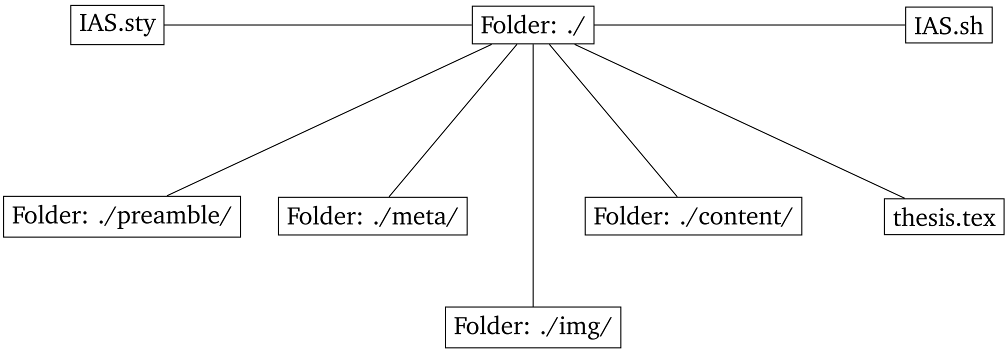

Figure 1.1.: The structure of the IAS Thesis LaTeX-Framework illustrated.

The **./preamble/** folder should contain content that needs to be processed in the preamble section, i.e. before `\begin{document}`, as the name suggests.

The **./meta/** folder should contain content that is not directly related to the topic of the thesis or summarizes content of the thesis, i.e. `abstract.tex` or `acknowledgements.tex`.

The **./img/** folder should solely contain images, e.g. `png`, `eps`, etc.

The **./content/** folder is the most important for the user eversince you stuff in all your content related files / chapters in here. You can `\input{yourFile}` afterwards in the `./thesis.tex` where the content variable is defined.

1.2.2 Commands & shortcuts

There are several shortcuts and commands you should memorize!

- Wrapper command for vectors

  `\cvec{}`

  Example: $\mathbf{v}$

- Wrapper command for matrices

  `\cmat{}`

  Example: $\mathbf{M}$

- Shortcut for `\textbf{}`

  `\bf{}`

  Example: **bold font**

- Shortcut for `\textit{}`

  `\it{}`

  Example: *italic font*

- Shortcut for `\underline{}`

  `\ul{}`

  Example: underline

- Shortcut for `\mathcal{}`

  `\mc{}`

  Example: $\mathcal{N}(\mu, \Sigma)$

1.2.3 Getting started with Glossaries

For a comprehensive guide to glossaries, you should read this article. There is also sample code given in `./preamble/glossary.tex` which you should have a look at as well!

2 Motivation

TODO

3 Foundations

- Bayesian optimization
- Global optimization
- Local optimization
- Gaussian Process Regression
- Hyper Parameter optimization Selecting good hyper parameters avoids numerical problems and enhances the efficiency ot our learning algorithm.
- Expected Improvement
- Thompson Sampling
- Standard kernel
- Trajectory kernel

Bayesian optimization
Figure 1D BO
Gaussian process
In this book we will be concerned with supervised learning, which is the problem of learning input-output mappings from empirical data (the training dataset). Depending on the characteristics of the output, this problem is known as either regression, for continuous outputs, or classification, when outputs are discrete. An example of a regression problem can be found in robotics, where we wish to learn the inverse dynamics of a robot arm. Here the task is to map from the state of the arm (given by the positions, velocities and accelerations of the joints) to the corresponding torques on the joints. Such a model can then be used to compute the torques needed to move the arm along a given trajectory.
In general we denote the input as x, and the output (or target) as y. The input is usually represented as a vector x as there are in general many input variables—in the handwritten digit recognition example one may have a 256-dimensional input obtained from a raster scan of a 16 × 16 image, and in the robot arm example there are three input measurements for each joint in the arm. The target y may either be continuous (as in the regression case) or discrete (as in the classification case). We have a dataset D of n observations, D = (x i , y i)|i = 1, . . . , n. Given this training data we wish to make predictions for new inputs x* that we have not seen in the training set. Thus it is clear that the problem at hand is inductive; we need to move from the finite training data D to a function f that makes predictions for all possible input values.
Trajectory kernel

3.1 Markov decision process

In our reinforcement learning problem we model the decision making as a Markov decision process. It consists of a tuple (S, A, P, P_0, R), holding all states $s \in S$, all actions $a \in A$, all state transitioning probabilities, and all corresponding rewards. Assume an agent which executes a policy θ for T time steps receiving a final reward $\bar{R}(\xi)$. We formulate the conditional probability of observing trajectory ξ given policy θ by

$$P(\xi|\theta) = P_0(s_0)\prod_{t=1}^{T} P(s_t|s_{t-1}, a_{t-1})P_\pi(a_{t-1}|s_{t-1}, \theta)$$

in which trajectory $\xi = (s_0, a_0, ..., s_{T-1}, a_{T-1}, s_T)$ contains the sequence of state, action tuples and $\theta \in \mathbb{R}^k$ a set of k policy parameters. $P_0(s_0)$ is the probability of starting in the initial state s_0. $P(s_t|s_{t-1}, a_{t-1})$ is the probability of transitioning from state s_{t-1} to s_t when action a_{t-1} is selected. And the stochastic mapping $P_\pi(a_{t-1}|s_{t-1}, \theta)$ is the probability for selecting the action a_{t-1} when in state s_{t-1} and executing the parametric policy θ. The final reward for a sampled trajectory

$$\bar{R}(\xi) = \sum_{t=1}^{T} R(s_{t-1}, a_{t-1}, s_t)$$

is the sum of all immediate rewards, given by an rewarding function $R(s_{t-1}, a_{t-1}, s_t)$. This rewarding function depends on the given environment. In general it rewards states we want to achieve and may penalize states we do not want our agent to be in.

3.2 Trajectory Kernel

Standard kernels like the squared exponential kernel, relate policies by measuring the difference between policy parameter values. Therefore policies with similar behavior but different parameters are not compared adequately. The behavior based Trajectory kernel fixes this, by relating policies to their resulting behavior. Hence our policy search gets more efficient by avoiding redundant search. To examine the difference between two policies θ_i and θ_j the Kullback Leibler divergence

$$D_{\mathrm{KL}}(P(\xi|\theta_i)||P(\xi|\theta_j)) = \int P(\xi|\theta_i) \log\left(\frac{P(\xi|\theta_i)}{P(\xi|\theta_j)}\right) d\xi$$

is applied to the respective policy-trajectory mappings $P(\xi|\theta_i)$ and $P(\xi|\theta_j)$. It measures how the two probability distributions diverge from another. When implementing the Kullback Leibler divergence we have discrete probability distributions. So the formula transforms to

$$D_{\mathrm{KL}}(P(\xi|\theta_i)||P(\xi|\theta_j)) = \sum_i P(\xi|\theta_i) \log \frac{P(\xi|\theta_i)}{P(\xi|\theta_j)}.$$

In general $D_{\mathrm{KL}}(P(\xi|\theta_i)||P(\xi|\theta_j))$ is not equal to $D_{\mathrm{KL}}(P(\xi|\theta_j)||P(\xi|\theta_i))$. So to achieve a symmetric distance measure we sum up the two divergences to

$$D(\theta_i, \theta_j) = D_{\mathrm{KL}}(P(\xi|\theta_i)||P(\xi|\theta_j)) + D_{\mathrm{KL}}(P(\xi|\theta_j)||P(\xi|\theta_i)).$$

To fulfill the requirements for a kernel the resulting matrix must be positive semi-definite and scalable. Therefore we exponentiate the negative of our distance matrix D. We also apply the hyper parameters σ_f and σ_l to make it scalable.

$$K(\theta_i, \theta_j) = \sigma_f exp(-\sigma_l D(\theta_i, \theta_j)).$$

4 Experiments

TODO

5 Results

TODO

6 Discussion

TODO

7 Outlook

TODO

Bibliography

[1] A. Kupcsik, M. Deisenroth, J. Peters, L. Ai Poh, V. Vadakkepat, and G. Neumann, “Model-based contextual policy search for data-efficient generalization of robot skills,” conditionally accepted.

[2] K. Muelling, A. Boularias, B. Mohler, B. Schoelkopf, and J. Peters, “Learning strategies in table tennis using inverse reinforcement learning,” accepted.

[3] C. Dann, G. Neumann, and J. Peters, “Policy evaluation with temporal differences: A survey and comparison,” no. March, pp. 809–883, 2014.

[4] T. Meyer, J. Peters, T. Zander, B. Schoelkopf, and M. Grosse-Wentrup, “Predicting motor learning performance from electroencephalographic data,” no. 1, 2014.

[5] B. Bocsi, L. Csato, and J. Peters, “Indirect robot model learning for tracking control,” 2014.

[6] H. Ben Amor, A. Saxena, N. Hudson, and J. Peters, “Special issue on autonomous grasping and manipulation,” 2014.

A Some Appendix

Use letters instead of numbers for the chapters.