LabMLISP: Lab Course Machine Learning in Signal Processing

Part I: Getting started

Prof. Veniamin Morgenshtern

Author: Kamal Nambiar

Your task in this lab

Your goal is to create a basic lane detection system using deep learning. You will be given a template of such system with algorithmic parts erased. At the minimum, your goal will be to re-create those parts. If you accomplish this task early, you will have the opportunity to participate in the research project by helping to extend the system and develop it further.

Here is a check list of what you should accomplish:

1. Setup your working environment. Use the instructions provided below in this document.
2. Develop a *road simulator*, that will produce artificial images of a road with lane markings depicted. For each image, the simulator must also produce a binary mask of the same resolution. The pixel in the mask is equal to 1, if the corresponding pixel in the image belongs to the road lane, and zero otherwise. The simulator and other parts of the system use the object oriented programming framework. If you are not comfortable with object oriented programming in Python, we provide a short tutorial for you and a link to a longer tutorial if needed. Please make sure you master the object oriented programming concepts while working on the simulator.
3. The lane detection system that you are building is based on deep learning. Read the "Deep learning" online book by Michael Nielsen. The book contains exercises. You will be given a list of recommended exercises to do. Out of that list you will be given several mandatory exercises. For the mandatory exercise you will need to timely submit well-written solutions, either in latex (for theoretical exercise) or as a Python notebook for programming exercises. *This is your first deliverable.*
4. Study *Pytorch*, a powerful and easy to use deep learning framework. We prepared a tutorial for you.
5. Create a neural network for lane detection in Pytorch.
    - Design a cost function.
    - Train the network using the data from the road simulator. Visualize the training process in Tensorboard.
    - Test the performance of the trained network on new images produced by the road simulator. Visualize the results. *This is your second deliverable.*
    - Test the performance of the trained network on real road images. *This is your third deliverable.*

6. Try to improve the performance of the system.
    - Use a collection of real world images that do not contain roads. Correspondingly, the mask for real images should consist of all zeros. Train the network on the mix of real world and artificial images. Test the performance of the network on real road images. Were you able to reduce the amount of false positives?
    - Implement data augmentations in order to improve performance. Retrain the network and test the results.
7. Take a real driving video, cut it into frames, apply your best lane detection network to each of the frames, overlay the result on top of the video. Create a new video with lanes detected. *This is your fourth deliverable.*

The lane detection results for real images do not need to be perfect, but they should be meaningful.

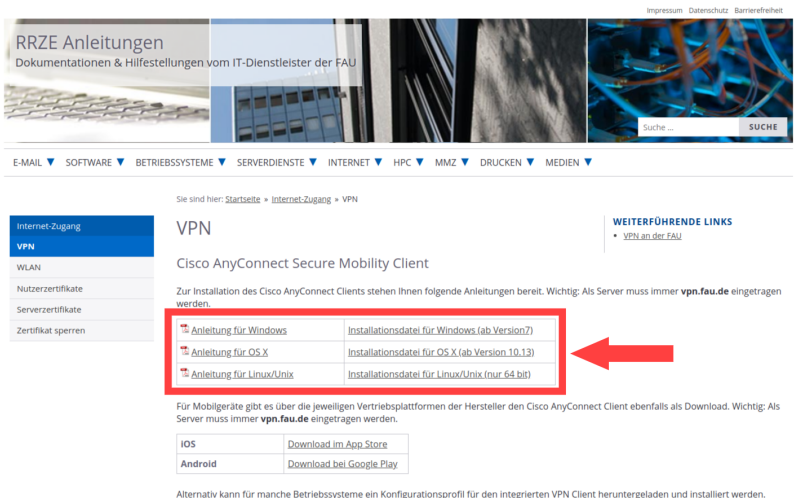
Once you are done with the plan above, you can help develop the system further. One interesting project might be to extend the road simulator to produce short artificial driving videos, and then use these artificial fragments to train a 3D network (x-y-time) that would work with a stack of images over time. This should significantly improve the performance.

Getting Started

In this lab we will be using the machine at *lms37-22.e-technik.uni-erlangen.de* to do our experiments.

Connecting to the university network

In order to access the remote machine, it is necessary to connect to the university network using a VPN. The official VPN client can be downloaded from: `https://www.anleitungen.rrze.fau.de/internet-zugang/vpn/`



Download the instructions ('Anleitung') and the installer according to your operating system, and follow the instructions to install the VPN Client.

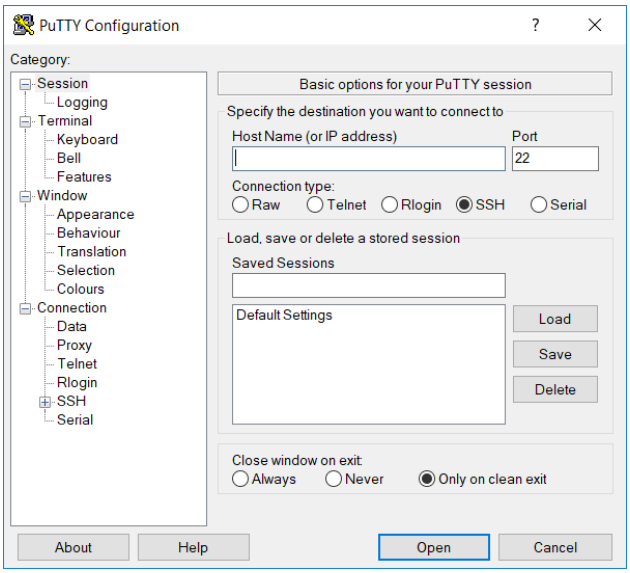
Without being connected through the VPN, you won't be able to connect to the lab machine. Instead, you will receive an error message like the following:

```
$ ssh mlisp-1@lms37-22.e-technik.uni-erlangen.de
ssh:  connect to host lms37-22.e-technik.uni-erlangen.de port 22:  Network is unreachable
```

Connecting to the remote system (Windows)

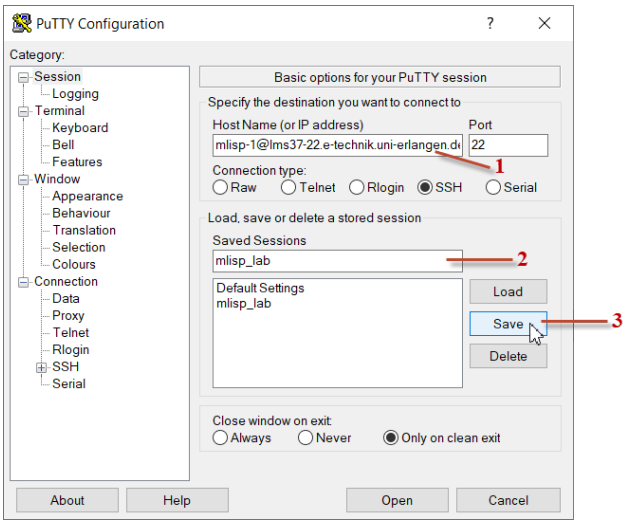
To connect to the lab machine, you will be using the ssh client 'PuTTY'. Download and install PuTTY using the installer (.msi file) from `https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html`.

A configuration window as shown below will open when you launch PuTTY.

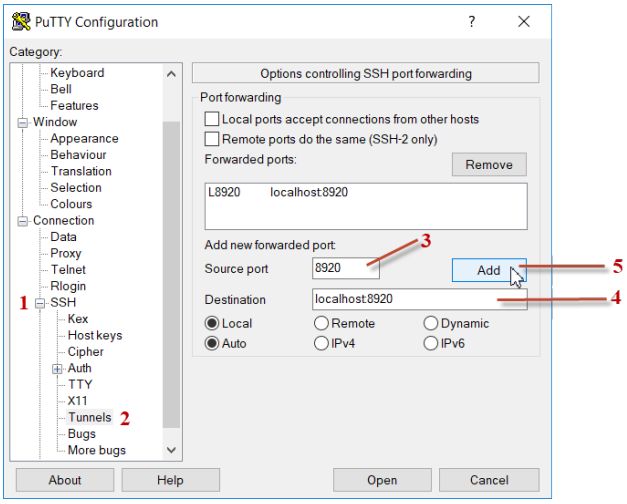


Enter the following details in the configuration window:

1. **Host Name (or IP address)** : username@lms37-22.e-technik.uni-erlangen.de
   Example: *mlisp-1@lms37-22.e-technik.uni-erlangen.de*
2. **Saved Sessions** : mlisp_lab
3. Hit the **Save** button on the right

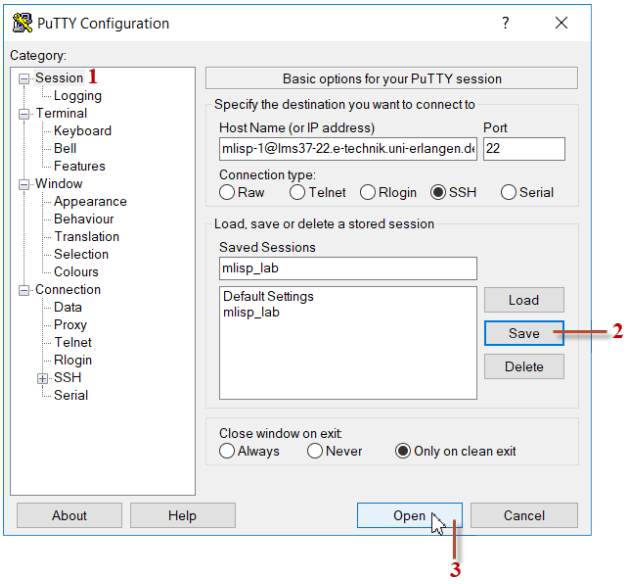


In this lab, you will be running your code in a jupyter notebook on the lab machine. To view the notebook on your local machine, we need to set up port forwarding.



1. On the left 'Category' panel, hit the + button next to SSH option
2. From the sub-menu, select the **Tunnels** option
3. **Source port** : Enter the port number provided to you
   Example: *8920*
4. **Destination** : localhost:port_number
   Example: *localhost:8920*
5. Hit the **Add** button on the right. Once this is completed, you will find an entry in the rectangular box above.

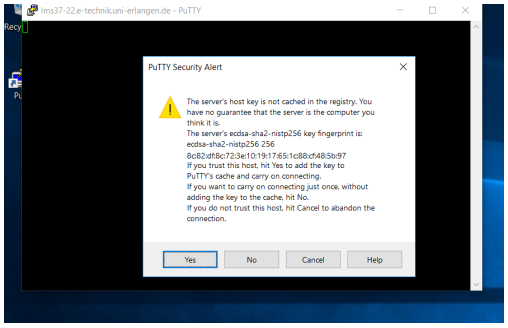
Save and Open the session

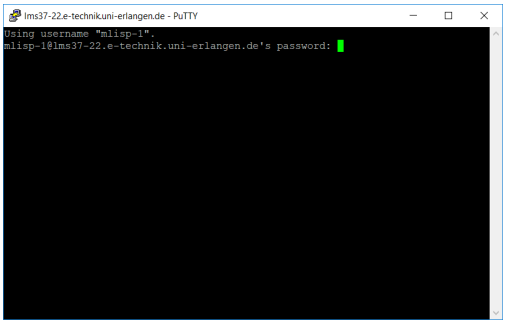


1. On the left 'Category' panel, hit the **Session** option
2. Hit the **Save** button on the right
3. Hit the **Open** button at the bottom

Once the connection is established, you will see a security alert window. Hit '**Yes**' and continue. Provide the password when prompted and continue with the rest of instructions from the **Environment Setup On Lab Machine** section

.





NOTE: The above instructions are only for initial setup of PuTTY ssh connection. The next time you launch PuTTY, you can load your saved settings by selecting the saved session name (mlisp_lab) from the list and hitting the **Load** button on the right. Subsequently hit the **Open** button to lauch new session.

Connecting to the remote system (Linux/Mac OS)

Open the terminal and login via ssh using the port number (`portnum`) and `username` provided to you:
`$ ssh -L localhost:portnum:localhost:portnum username@lms37-22.e-technik.uni-erlangen.de`

Example:
`$ ssh -L localhost:8920:localhost:8920 mlisp-1@lms37-22.e-technik.uni-erlangen.de`

Environment Setup On Lab Machine

NOTE: The below instructions are for the initial setup of your working environment on the lab machine. The next time you login to this machine, we can navigate to your working directory(created below) and continue with your work.

- Create your *working directory* and change your current directory to this newly created directory.
  `$ mkdir labmlisp`

  `$ cd labmlisp`

- Copy files from the shared folder `/SHARED/DATA/mlisp-lab/`
  NOTE: Do not forget the dot '.' at the end of the statement. If you are copy-pasting the commands from this document into the terminal, pay attention to all special characters in the command (like the underscore '_').
  `$ cp ~/SHARED/DATA/mlisp-lab/ps1_dependencies/*.txt .`

- All python modules required for the lab can be installed in two stages. Create conda environment and install all modules mentioned in the `spec_file.txt` using:
  `$ conda create --name lab --file spec_file.txt`

- Activate the above created conda environment
  `$ source activate lab`

- Install the remaining dependencies from `requirements.txt` using pip command.
  `$ pip install -r requirements.txt`

Optional: tmux

`tmux` allows you to maintain several terminal windows inside an ssh session. After you connect to remote machine through ssh, run:
`$ tmux`

Here are some useful hot keys that you may use in tmux.

- Create a new window : press Ctrl b , release it and then press c .
- Switch between different windows : press Ctrl b , release it and then press n .
  You can now use one of the windows to run a Jupyter Notebook server (next section) and another window for managing files, for example.
- Close tmux : press Ctrl b , release it and then press d .

More information on how to use tmux can be found in the cheat sheet published on course web page and online.

Very important: tmux sessions are persistent and they will continue to run when you go home. When you come next time, please **do not create a new tmux session**! Instead, type
`$ tmux attach`
to get back to your previously created tmux session.

Jupyter Notebook

In this lab, we recommend the students to use Jupyter Notebook for code development. You will be provided with a few notebooks that will help you test your implementation. If you have not worked with Jupyter Notebook before, please take the time to familiarize yourself with Jupyter Notebook using resources available online.

- Before you launch Jupyter Notebook, make sure that you are in working directory. You can use the `pwd` command to verify if you are not sure. Also be sure that the conda environment that you had created earlier is activated. You would see your conda environment name (lab) before the prompt, if this is done correctly.
- Launch Jupyter Notebook using the port number (`portnum`) and GPU number (`gpunum`) provided to you along with your login credentials.
  `$ CUDA_VISIBLE_DEVICES=gpunum jupyter notebook --no-browser --port=portnum`

  Example:
  `$ CUDA_VISIBLE_DEVICES=7 jupyter notebook --no-browser --port=8920`
- The previous command will print a long web link, which you should now open in the web browser on your local machine.

- You can use upload button on the top right corner of the page to copy all the lab materials that you may require for your code development to your working directory.
- Once you have completed your work and are leaving for the day, close Jupyter by entering Ctrl c in the terminal.
- Jupyter has many useful hot keys, for example the hot key for commenting multiple lines of code is Ctrl / . Note that this only works on English keyboard. Here is a useful cheat sheet: `https://www.cheatography.com/weidadeyue/cheat-sheets/jupyter-notebook/pdf_bw/`
- If you insist on using German keyboard layout, here is a tip: `https://stackoverflow.com/questions/26504796/how-can-i-block-comment-code-in-an-ipython-notebook-with-a-german-keyb`

Miscellaneous

You are provided with a very powerful machine with multiple graphics cards for your experiments. If you are interested to see the status of the GPU, memory consumed by your processes, etc, you can run the below command.

```
$ nvidia-smi
```