Advice for a Successful Research Project

Jake Lever - University of Glasgow

Disclaimer: These are suggestions from my experience supervising numerous undergraduate/MSc projects and are not official policy from the school. They may not map perfectly to your project or experience so take from the suggestions what you can.

Introduction

Projects are a great opportunity to take some of what you learned in your courses and go much much deeper into a particular problem and set of methods. The first key concept is that **this is your project**. I will provide guidance and feedback, but I won't chase you for making progress. It is down to you to dig into the project, read the papers, write the code, check it is working and write up the report. It will also be down to you to set yourself deadlines and figure out your own schedule.

Types of Projects

There are broadly two types of projects that undergraduate and MSc project students may undertake: a research project (often focused on a machine-learning pipeline) and an engineering project (building a usable tool/product). Some undergraduates and IT MSc students do engineering projects but most of my students do research projects on machine learning problems. Some projects can combine bits of both types, but it is somewhat simpler to stick to one type for a more straightforward project.

Research projects focussed on machine learning involve finding a machine learning problem with an associated dataset, e.g. a classification problem with a dataset that has annotations. The dataset can then be used to benchmark some ideas. You will first do a literature review to see how others have approached the problem and propose your approach. Your approach will likely take a few ideas from other works and see how they perform for your dataset and problem. Your evaluation will use standard machine learning metrics (e.g. precision, recall, F1 for a classification problem).

Engineering projects involve identifying and understanding a problem, examining how others have approached it (e.g. other products or academic research) and doing some form of requirements capture for the problem. This will help decide what features need to implement and therefore what type of technology to use to build the product (e.g. web app, mobile app, etc). You'll then design the user interface (possibly with wireframes), build it and do some form of user study to evaluate if you have succeeded in your goals.

This document focuses on research projects on machine learning.

Literature Survey/Review

The purpose of a literature review is to see how others have approached the problem so you can decide how you are going to do it.

Google Scholar is my normal tool for searching papers. If you find a recent relevant paper, read its background section in detail and see what papers it cites. If there are older papers, you can find them on Google Scholar and see what other papers cite them. Click the “Cited by X” link under the paper title to see that list. Some machine learning problems also have multiple names, so reading a few more papers may give you more ideas of what keywords to use when searching. Also, if the dataset for your project has an associated paper, see what other work cites it as they may have used the dataset.

It’s a good idea to keep track of what papers you’ve read and what you think. Note, you don’t have to read every paper you come across in detail. At least read the abstract, and if it is relevant give it a skim. Pick a smaller number of highly relevant papers to read in detail. It can also help to read a paper a few times to fully understand what’s going on. Paper management tools like Zotero can help you index papers you’ve looked at. After reading a paper, it’s a good idea to write a couple sentences about the relevant papers that you may cite in your report later, as you will forget what the paper is about. Having a set of sentences about the relevant papers can make it much easier to put together the literature review portion of your report at the end.

Other sources of papers could include the leaderboards on paperswithcode.com, shared task/community competitions on your dataset, blog posts, review articles in the area and Twitter/X.

A good literature review looks for patterns in what other people have tried for the problem. See if you can roughly categorise the different approaches. For instance, different approaches for a problem could try “data augmentation” or “model ensembling”. This makes for a much clearer and nicer written literature review that talks about other works in clear groups as opposed to something that reads more like a list of prior works.

Research Questions

A good project asks some kind of question (or preferably multiple questions). “Does doing X make the system do better or worse?” These should be explicitly written out in your report (potentially at the end of your Background Chapter) and then referenced as section headings in your Results chapter.

An important note is that you are not graded on the success of your machine learning approach. It may perform much worse than a baseline approach. But as long as the idea seemed reasonable, then you’ll be graded well. Admittedly, if the performance is zero or perfect, that suggests something has gone wrong.

Implementation

Developing in Python Notebooks/Colab is a reasonable place to start for Python projects. Many students do their entire projects inside notebooks - which may not always be the best idea. Consider when it may be a good idea to transition to Python (.py) files and use libraries such as argparse to create standalone programs that can be run through the terminal.

It is definitely a good idea to use source control such as GitHub. Notebooks do not work brilliantly with source control (as many lines change every time you run the notebook). Hence Python files are often the better way to go. I also recommend putting your project on GitHub with a nice README so that you can share it on your CV. This will be very useful for future job applications.

Working with limited compute

Your project may require more compute (e.g. for fine-tuning a transformer model). Typically, project students don't have a lot of access to compute infrastructure for their projects. You will need to scale your ideas and approaches appropriately to fit within the appropriate scale for a student project.

Some ideas that may be feasible:

- Reduce the size of the dataset (number of samples)
- Put other constraints such as reducing the number of labels to be predicted (for a multi-label/multi-class problem)
- Use free GPU time on Colab and Kaggle where appropriate
- Select less compute-heavy methods (e.g. reduce the need to fine-tune transformer models)
- Use the new breed of quantized transformer models (e.g. LLAMA2) that can be run on small devices and CPUs

Evaluation

A large part of the final grade on the report is on the evaluation of your project. Essentially, after building a thing, did you succeed?

One of the first decisions is what metrics will be used for evaluation (e.g. precision, recall, F1-score). It's often a good idea to use more than one metric so that you are looking at the performance from more than one angle. Your literature survey should find other approaches to the problem and provide some ideas of the metrics commonly used for the problem.

Then you can decide how to present the results. While it may be good to check confusion matrices, it's not always useful to put them all in your report. Be picky with which results you put them in and how you show them. Simpler is better and barcharts/tables of results can be enough to get your point across.

Notably we don't expect state-of-the-art results. That would be unreasonable for an L4 project. The actual results may be terrible - and that's broadly okay. Do look out for 0% or 100% performance which suggests something has gone wrong with the implementation. Also beware of classifiers that only predict one class (e.g. always predicts positive).

The most important part of evaluation is not the results themselves but your discussion of them. You should talk about what you think may be happening and how the results relate to answering your research questions. You can hypothesise a little about possible causes of interesting behaviour. The key thing is not to simply state your results but discuss what they mean.

Weekly Meetings

Normally I meet students once a week for 30 minutes. At the beginning of the project, I will likely lead the meetings and provide an introduction to the problem and hopefully convince you why it's exciting. In later meetings, it will be up to you to drive the meetings and decide what we should talk about.

The exact format of the meetings is up to you. Some students write an agenda and minutes, write weekly updates beforehand or bring slides to talk through. It's up to you. I do recommend bringing a list of questions to discuss and minutes are a good idea for tracking progress. I encourage you to bring a notepad or laptop if you want to take notes. You may also want your laptop if you want to show or discuss any specific work. I am also fine with recording meetings if desired.

Writing Your Project Report

Many of the templates that you are provided with are for engineering-style projects that have requirements capture, a separate design and implementation phase. You can disregard that structure and follow the broad chapter structure outlined below. Don't feel tied to this and it is only a suggestion.

1. Introduction
    - Here you provide a brief introduction of what the problem is and why it is important. Keep it high-level. Remember that your reader may have no background in this area
    - Talk broadly about the methods that will be explored for the problem
2. Background
    - This chapter typically has three parts:
        i. Provide a deep background on the problem and explain some of the concepts. It is essential that the reader knows what you are trying to solve. A figure giving an example of the problem can be invaluable here
        ii. The literature survey: How have others approached the problem? See the notes above about grouping the survey into sections about the broad types of approaches that others have taken.
        iii. Given what you've read in the literature survey, what are the opportunities to try out some ideas? Here you should explicitly list out your research questions as bullet points - think RQ1, RQ2, etc. You can then refer back to them in the Results chapter.
3. Methods *(sometimes called Implementation or Design & Implementation)*
    - Here you describe how you are going to undertake your experiments
    - You may want to describe the dataset that you'll be using for the project here. It may also be reasonable to talk about it in the background section if it makes sense.
    - A figure showing a flow-chart of the main processes of the project (or some other kind of overview figure) can be beneficial here.
    - The main part of this chapter is working through the steps of your project with appropriate sections and subsections.

    - The general idea is to describe how you did the project in enough detail that someone else (with some knowledge of the area) could re-implement your project
    - Things like how you did hyperparameter tuning also goes in here
4. Results *(sometimes called Evaluation)*
    - This section should refer directly back to your research questions outlined in Chapter 2
    - Have a section for each research question. Include a results figure or table and then discuss what it means
5. Conclusion
    - Provide a nice summary of what your results mean together
    - Discuss any limitations that you face in the project or limitations on how the results may be interpreted (e.g. they may not generalise)
    - Discuss any future directions that you would take (given more time)

Tips for Writing

- My policy is to read a single draft and give feedback for each student. You can provide drafts chapter-by-chapter or the whole thing. Just remember to leave enough time so that I can read it and get you feedback. Ideally a couple weeks before the deadline.
- Your dissertation report will be marked by another academic from the School who may not know much about the area. Be very gentle when you introduce the topic. Clearly describe the motivation and exactly what the problem is.
- Avoid restating numbers from results tables in text. Instead talk about what the results mean, or describe trends that may not be immediately obvious
- Simpler is always better. Simple tables and simple graphs (e.g. barcharts) will make your message clearer.
- Avoid including screenshots of code unless it really provides a useful explanation of the implementation that an alternative doesn't.
- You may end up with many many graphs during your analysis. Don't include them all. Each figure should have a clear message (e.g. method A is better than method B). Make it simple and clear.
- Make use of topic sentences when writing. The basic idea is that the first sentence of each paragraph should state the main
- Writing is a learned skill and we all start off rubbish at it. Editing is key. Give yourself enough time to reread parts of your report and redraft.
- Inverse Chekhov's Gun - Don't introduce new methods or ideas in a later chapter that have not been suitably introduced and set up earlier

The And-But-Therefore approach

This method is invaluable for writing the abstract and introduction of your report (and many future documents). It sets up the problem, the motivation and broadly your approach. The points below outline the three parts with example text.

- The AND: A few sentences about the current situation that is relevant to your project.

    - e.g. Biomedical machine learning methods promise to revolutionise how diseases are diagnosed and usher in an era of precision medicine. Transformer-based methods are able to integrate knowledge from biomedical texts for this task.
- The BUT: A sentence describing what the problem is. It often starts with “However”. The problem should be fairly severe and can be emphasised with numbers (e.g. $$$ costs, health impact, etc) if available.
    - e.g. However, these methods rely on large amounts of annotated data which is infeasible given the high cost of biomedical expert annotators.
- The THEREFORE: A sentence or two that describes your proposal to solve the problem
    - e.g. We propose a new method that uses distant supervision to generate high quality annotations with minimal user input.

You can use it directly for an abstract and in a longer form for an introduction.

Presentations

You may be asked to present on your research project. Here are a few tips for good presentations.

- The most important thing is: **think about your audience**. Are they experts or new to the topic? This will help you tailor the talk to them.
- It’s almost always better to go simpler than more complex
- Clearly describing the motivation (the why) and the exact problem (the what) is key. Use examples. Perhaps walk the audience through a simple example.
- Think about how much text is on your slide.
    - Do you expect the audience to read all of the text?
    - Avoid simply reading off the text from the slide
    - Using no text is an interesting challenge to try (maybe only once)
    - You do want to keep text fairly minimal though
- Think about why you have slides at all
    - Slides are probably a bit of a crutch - to remind you what to talk about. We all use them for that
    - But they should also support your talk. Text and images on the slides can help the audience understand your point more.
- Practise the timing of your talk. You should know roughly how long the talk should be. Use a timer and see if you are too long or short. If you are too long, cut content. Do not just try to speak faster. If you are short, consider explaining one of your slides in more detail before adding more content
- Learn the transitions between slides. The most common trip up when presenting is that you can’t remember when to stop speaking on one slide and what the next slide is on.
- Speak out to your audience. Practice looking outwards to your audience. It may be useful to set yourself up in a good position so that you can glance down at a screen (instead of backwards at the slides projected on a wall). Try to make eye contact with a few audience members

Common Questions

Here are a few common questions that come up related to projects.

Q: I really don't know where to get started coming up with a new solution for my problem. What should I do?

A: One approach is to select a paper that has already approached the problem and re-implement it. You don't need a completely novel method. You can try out a few slight variations of their approach as you implement it.

Q: How do I come up with research questions?

A: You are unlikely to have a full set of research questions when you start implementing your project. Generally, you will have to make some arbitrary decisions as you implement (e.g. should I use library A or library B for this part). That could be a research question: "what effect does library A or library B have on performance?"

Q: Can I use existing code to solve my problem?

A: You can use existing modules, libraries and example code out there in your project. However, if you have found code that does your entire project for you with very minimal coding yourself, your project may be too simple. Let me know and we can discuss it.

Q: Is it possible to publish a paper from my project?

A: Yes, though it is rare. Some broad tips towards that:

- The literature survey is a critical part to try to find some new ideas to try out. You'll need to read widely to understand what approaches have been taken and what methods you might try.
- You will want to compare your approach to other reasonable approaches in the literature (and there may be an obvious state-of-the-art approach). You may need to implement that yourself or fight with someone else's code
- It is likely that your first approach won't be immediately publishable and that you'll need to iterate and dive deeper into what's working and what's not. This involves some solid time management to get a baseline and your ideas implemented fast enough that you have time to iterate and try things out.