Practical Tips for Final Projects

Plan

1. Final project types and details; assessment revisited
2. Finding research topics; a couple of examples
3. Finding data
4. Doing your research
5. Presenting your results and evaluation

1. Course work and grading

- Participation (Piazza contribution): 10%
- Project (2-4 people):
  - Proposal (10.3.2020, 8am): 20%
  - Midterm report (31.03.2020, 8am): 20%
  - Final report (28.04.2020, 8am): 40%
  - Presentation (28.04.2020): 10%

The Project

- Propose a custom research project, which we must approve
- You work in teams of 2-4
- Larger team project or a project for multiple classes should be larger and often involve exploring more tasks
- You have to use Python (PyTorch) and focus on a NLP task using deep learning

Why do a research project?

- Opportunity to apply the concepts learnt in class creatively
    - Helps to understand material more deeply!
- Designing and working on a unique project in a team is what you do in a workplace
    - Project helps you prepare for that
- Boost your CV and add to your portfolio
- Learn how to do research
    - Helps with your other research projects and dissertations!

Perfect project

- There is no “perfect project”
- Be ambitious but the key aspect of the project is your learning experience
- Don’t pick something that is too easy. But also don’t pick something that’s outside the scope of the class
- You are not graded on how exciting is your proposal but on whether you achieve the objectives of the proposal, midterm and final reports, and presentation.

2. Finding Research Topics

- Two basic starting points (pretty much for all of science):
  - [Nails] Start with a (domain) problem of interest and try to find good/better ways to address it than are currently known/used
  - [Hammers] Start with a technical approach of interest, and work out good ways to extend or improve it or new ways to apply it

Project types

This is not an exhaustive list, but most projects are one of

- Find an application/task of interest and explore how to approach/solve it effectively, usually applying an existing model
- Implement a complex modelling architecture and demonstrate its performance on some data
- Come up with a new or variant model and explore its empirical success
- Analysis project. Analyse the behaviour of a model: how it represents substantive knowledge or what kinds of phenomena it can handle or errors that it makes
- Rare theoretical project: show some interesting, non-trivial properties of a model type, data, or a data representation

How to find an interesting place to start?

- You could browse recent publications at any of the top venues: ACL, EMNLP, TACL, NAACL, EACL, NIPS, ICLR, ICML. (This list are not exhaustive!)

- Try a keyword search at:
  - http://arxiv.org/
  - http://scholar.google.com
  - http://dl.acm.org/

- Look at publications from top research groups (e.g. Stanford NLP group) https://nlp.stanford.edu/pubs/

- Many top ML/NLP venues will have workshops on more applied topics that contain recent research on that task or problem. Here is an example of a workshop at ICML on the contribution AI/ML can make to climate change research: https://www.climatechange.ai/ICML2019_workshop.html

How to find an interesting place to start?

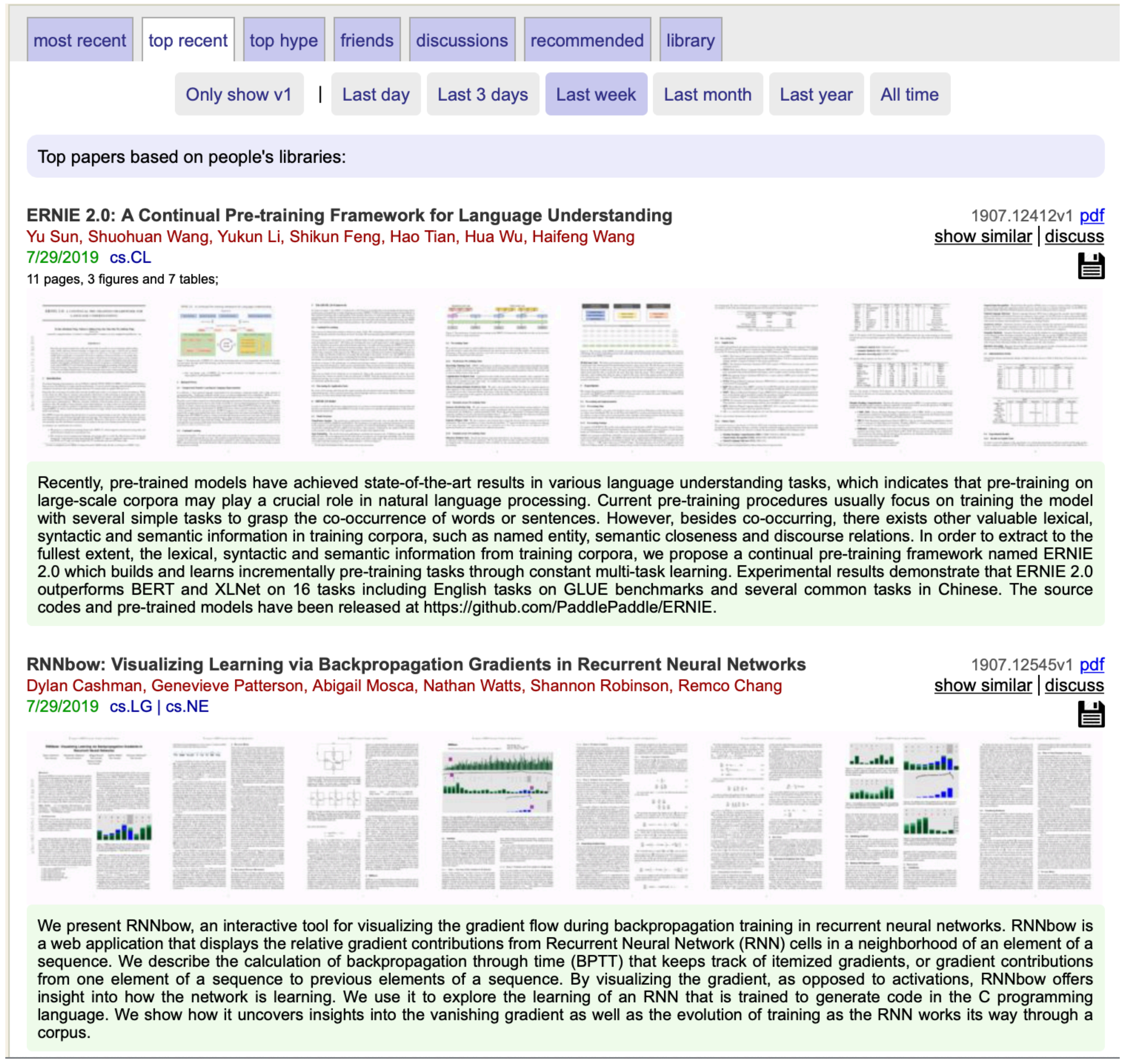
- Even better: look for an interesting problem in the world!

Faculty projects

- We posted a set of projects on Piazza that we think would be interesting to work on and could potentially result in a publication.

How to find an interesting place to start?

- Arxiv Sanity Preserver by Anrej Karpathy (head of AI at Tesla):
- http://www.arxiv-sanity.com



Top papers based on people's libraries:

ERNIE 2.0: A Continual Pre-training Framework for Language Understanding
Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, Haifeng Wang
7/29/2019 cs.CL
11 pages, 3 figures and 7 tables;
1907.12412v1 pdf
show similar | discuss

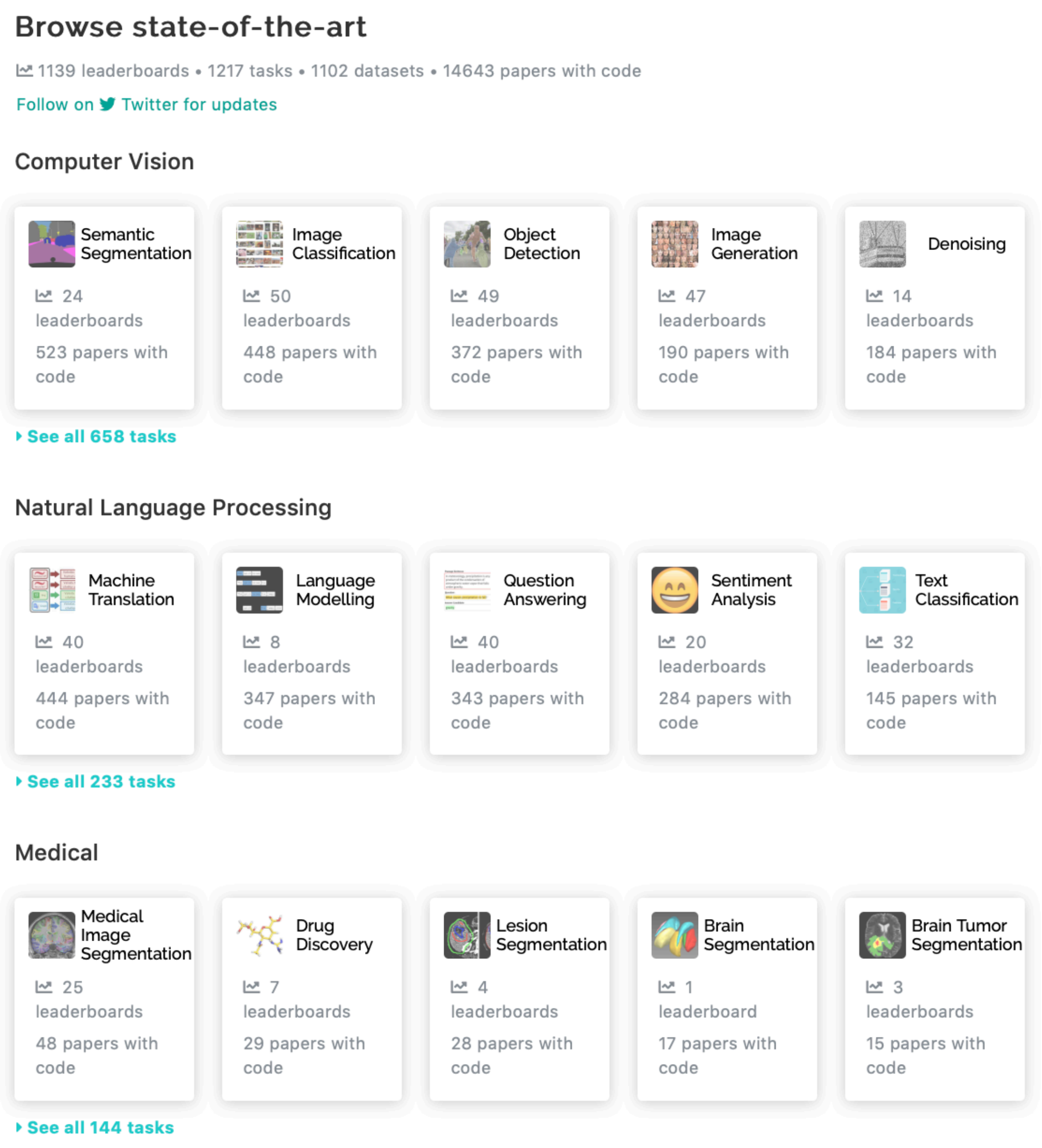
Recently, pre-trained models have achieved state-of-the-art results in various language understanding tasks, which indicates that pre-training on large-scale corpora may play a crucial role in natural language processing. Current pre-training procedures usually focus on training the model with several simple tasks to grasp the co-occurrence of words or sentences. However, besides co-occurring, there exists other valuable lexical, syntactic and semantic information in training corpora, such as named entity, semantic closeness and discourse relations. In order to extract to the fullest extent, the lexical, syntactic and semantic information from training corpora, we propose a continual pre-training framework named ERNIE 2.0 which builds and learns incrementally pre-training tasks through constant multi-task learning. Experimental results demonstrate that ERNIE 2.0 outperforms BERT and XLNet on 16 tasks including English tasks on GLUE benchmarks and several common tasks in Chinese. The source codes and pre-trained models have been released at https://github.com/PaddlePaddle/ERNIE.

RNNbow: Visualizing Learning via Backpropagation Gradients in Recurrent Neural Networks
Dylan Cashman, Genevieve Patterson, Abigail Mosca, Nathan Watts, Shannon Robinson, Remco Chang
7/29/2019 cs.LG | cs.NE
1907.12545v1 pdf
show similar | discuss

We present RNNbow, an interactive tool for visualizing the gradient flow during backpropagation training in recurrent neural networks. RNNbow is a web application that displays the relative gradient contributions from Recurrent Neural Network (RNN) cells in a neighborhood of an element of a sequence. We describe the calculation of backpropagation through time (BPTT) that keeps track of itemized gradients, or gradient contributions from one element of a sequence to previous elements of a sequence. By visualizing the gradient, as opposed to activations, RNNbow offers insight into how the network is learning. We use it to explore the learning of an RNN that is trained to generate code in the C programming language. We show how it uncovers insights into the vanishing gradient as well as the evolution of training as the RNN works its way through a corpus.

Want to beat the state of the art on something?

- Great new site
- Though not always up to date
- https://paperswithcode.com/sota



Browse state-of-the-art

1139 leaderboards • 1217 tasks • 1102 datasets • 14643 papers with code

Follow on Twitter for updates

Computer Vision

Task	Leaderboards	Papers with code
Semantic Segmentation	24 leaderboards	523 papers with code
Image Classification	50 leaderboards	448 papers with code
Object Detection	49 leaderboards	372 papers with code
Image Generation	47 leaderboards	190 papers with code
Denoising	14 leaderboards	184 papers with code

See all 658 tasks

Natural Language Processing

Task	Leaderboards	Papers with code
Machine Translation	40 leaderboards	444 papers with code
Language Modelling	8 leaderboards	347 papers with code
Question Answering	40 leaderboards	343 papers with code
Sentiment Analysis	20 leaderboards	284 papers with code
Text Classification	32 leaderboards	145 papers with code

See all 233 tasks

Medical

Task	Leaderboards	Papers with code
Medical Image Segmentation	25 leaderboards	48 papers with code
Drug Discovery	7 leaderboards	29 papers with code
Lesion Segmentation	4 leaderboards	28 papers with code
Brain Segmentation	1 leaderboard	17 papers with code
Brain Tumor Segmentation	3 leaderboards	15 papers with code

See all 144 tasks

Finding a topic

- Turing award winner Ed Feigenbaum says to follow the advice of his advisor, AI pioneer, and Turing and Nobel prize winner Herb Simon:

"If you see a research area where many people are working, go somewhere else."

Must-haves

- Suitable data
- Feasible task
- Automatic evaluation metric
- NLP with deep learning central to the project

3. Finding data

- Some people collect their own data for a project
  - You may have a project that uses “unsupervised” data
  - You can annotate a small amount of data
  - You can find a website that effectively provides annotations, such as likes, stars, ratings, etc.: Let’s you learn about real world challenges of applying ML/NLP!

- Some people have existing data from a research project or company
  - Fine to use providing you can provide data samples for submission, report, etc.

- Most people make use an existing, curated dataset built by previous researchers
  - You get a fast start and there is obvious prior work and baselines

Finding data

There are lots of publicly-available datasets on the web. Here are some useful resources to find datasets:

- Wikipedia has a list of machine learning text datasets, tabulated with useful information such as dataset size. https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research
- Kaggle has many datasets, though some of them are too small for NLP DL. https://www.kaggle.com/datasets
- Datahub has lots of datasets, though not all of it is Machine Learning focused https://datahub.io/collections
- Microsoft Research has a collection of: https://msropendata.com
- Google Dataset Search: https://toolbox.google.com/datasetsearch
- A script to search arXiv papers for a keyword, and extract important information such as performance metrics on a task: https://huyenchip.com/2018/10/04/sotawhat.html
- A collection of links to more collections of links to datasets: http://kevinchai.net/datasets
- A collection of papers with code on many tasks: https://paperswithcode.com/sota
- Many social science datasets are deposited (and indeed this is a requirement for many journals!) on the Harvard Dataverse: https://dataverse.harvard.edu

4. Doing your project

1. Define Task
2. Define dataset
    - Search for academic datasets: they already have baselines
    - Define your own data (harder, need new baselines)
        - Allows connection to your research
        - A fresh problem provides fresh opportunities!
        - Be creative: World Bank, IMF, Twitter, Blogs, news, etc. There are lots of neat websites which provide creative opportunities for new tasks

3. Dataset hygiene: right from the start, separate off dev, validation, and test splits (more detail below)

4. Define your metric(s)
   - search online for well established metrics on this task
   - human evaluation is often still better, and you may be able to do a small scale human evaluation

5. Establish a baseline
   - Implement the simplest model first (often linear regression or in NLP it can be a logistic regression on unigrams and bigrams or averaging word vectors)
   - Compute metrics on train AND dev
   - Analyse errors
   - If metrics are amazing and no errors: Done! Problem was too easy. Need to restart.

6. Implement existing complex model
   - compute metric on train and dev
   - analyse output and errors
   - minimum bar for this class

7. Always be close to your data! (Except for the final test set!)
    - visualise the dataset
    - collect summary statistics
    - look at errors
    - analyse how different hyper-parameters affect performance

8. Try out different models and model variants. Aim to iterate quickly via having a good experimental setup

Pots of data

- Many publicly available datasets are released with a **train/dev/test** structure. We're all on the honour system to do test-set runs only when development is complete.

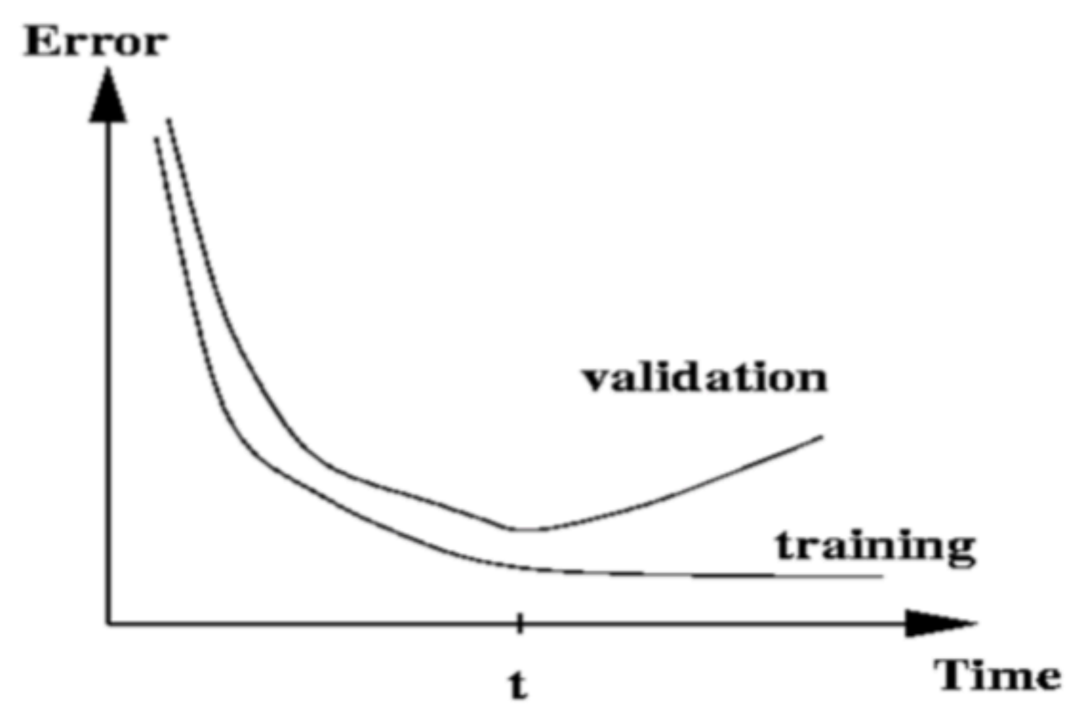
- Splits like this presuppose a fairly large dataset. If your dataset is small then either explore cross-validation as an alternative approach or think about a different problem (and dataset).

- If there is no dev set or you want a separate tune set, then you create one by splitting the training data, though you have to weigh its size/usefulness against the reduction in train-set size.

- Having a fixed test set ensures that all systems are assessed against the same gold data. This is generally good, but it is problematic where the test set turns out to have unusual properties that distort progress on the task.

Training models and pots of data

- When training, models **overfit** to what you are training on
  - The model correctly describes what happened to occur in particular data you trained on, but the patterns are not general enough patterns to be likely to apply to new data
- The way to monitor and avoid problematic overfitting is using **independent** validation and test sets ...



Training models and pots of data

- You build (estimate/train) a model on a **training set.**

- Often, you then set further hyper-parameters on another, independent set of data, the **tuning set**
    - The tuning set is the training set for the hyper-parameters!

- You measure progress as you go on a **dev set** (development test set or validation set)
    - If you do that a lot you overfit to the dev set so it can be good to have a second dev set, the **dev2** set

- **Only at the end**, you evaluate and present final numbers on a **test set**

    - Use the final test set **extremely** few times … ideally only once

Training models and pots of data

- The **train, tune, dev,** and **test** sets need to be completely distinct
- It is invalid to test on material you have trained on
    - You will get a falsely good performance. We usually overfit on train
- You need an independent tuning set
    - The hyper-parameters won't be set right if tune is same as train
- If you keep running on the same evaluation set, you begin to overfit to that evaluation set
    - Effectively you are "training" on the evaluation set...you are learning things that do and don't work on that particular eval set and using the info
- To get a valid measure of system performance you need another untrained on, **independent** test set ... hence dev2 and final test

Getting your machine learning model to train

- Start with a positive attitude!
  - ML models want to learn!
    - If the model isn't learning, you're doing something to prevent it from learning successfully

- Realise the grim reality:
  - There are lots of things that can cause ML models to not learn at all or to not learn very well
    - Finding and fixing them ("debugging and tuning") can often take more time than implementing your model

- It's hard to work out what these things are
  - But experience, experimental care, and rules of thumb help!

Experimental strategy

- Work incrementally!
- Start with a very simple model and get it to work
- Add bells and whistles one-by-one and get the model working for each of them (or abandon them)
- Initially run on a tiny amount of data
    - You will see bugs much more easily on a tiny dataset
    - Something like 8 examples is good
    - Often synthetic data is useful for this
    - Make sure you can get 100% on this data
        - Otherwise your model is definitely either not powerful enough or it is broken

Experimental strategy

- Run your model on a large dataset
  - it should still score close to 100% on the training data after optimisation
    - Otherwise, you probably want to consider a more powerful model
    - Overfitting to training data is **not** something to be scared of when doing machine learning
- But, still, you want good generalisation performance
  - Regularise your model until it doesn't overfit on dev data
    - Strategies like L2 regularisation can be useful

Details matter!

- Look at your data, collect summary statistics
- Look at your model’s outputs, do error analysis
- Tuning hyper-parameters is really important to almost all of the successes in machine learning

5. Presentation

- A poster or video
- Give the viewer who is not familiar with your work a quick and easy to understand overview.
- Highlight only the most important and interesting results.
- Visualisations help!

Poster or video content

- Problem: Briefly explain what problem you are tackling, why it’s important, and what the existing approaches are (you might also mention the limitations of these approaches).
- Data/Task: Briefly explain (or better, give examples of) the data and/or task.
- Proposed method: Describe your main methods/techniques/models. Diagrams are generally better than text, and equations should be used sparingly (if at all). Highlight the core idea of your techniques.
- Results: Present your most important results. Tables containing many numbers are overwhelming. Be selective and choose just the results that convey the story you’re telling. Make it clear what the evaluation metrics are, and what’s being compared.
- Analysis: Show any plots, diagrams, examples and visualisations to provide interesting analysis. Make it clear what the reader should conclude from each figure.
- Conclusions: Briefly draw some main conclusions from your work. If you wish, you can outline future work.
- References: Posters typically have just a few references (e.g. 1-3), just for the papers that are core to the work.

Poster or video content

- Aim to make the content visual (both poster and video) rather than text heavy
- Clean and minimal is better than cluttered
- Intuitive rather than complicated!

Poster-specific instructions

- At the poster session you will be supplied with a standard posterboard (width 145cm, height 115cm), an easel, and push pins.
- Your poster will look best if you print it on one large sheet of paper (e.g. A0 size), but if you assemble your poster out of smaller sheets that's OK too.
- Some advice for making good scientific posters: https://courses.physics.illinois.edu/phys596/fa2013/Lectures/ScientificPosterTips_FA12.pdf

Video-specific instructions

- Your video should be a 3-5 minute overview of your work, covering the content above.
- You could think of it as similar to a ‘lightning talk’ given at a research conference.
- The recommended and easiest way to make your video is to make some slides, and then to record a screencast (i.e. you record the screen and record yourself talking – your face doesn’t appear).
- There’s lots of free software to do this – search online for ‘screencast’.
- You can make a fancier video if you like, but a screencast is perfectly sufficient.

Presentation grading

- Each group tends to the poster stand and answers questions from your colleagues
- Three components to the grade: best presentation, most creative project, and best visualisations
- Voting determines the grade: score card for each presentation
- Everybody fills out a card per presentation scoring them on 1-10 scale. You do not score your own group.
- Average score is the mark.
- See Presentation instructions for more details.

Good luck!