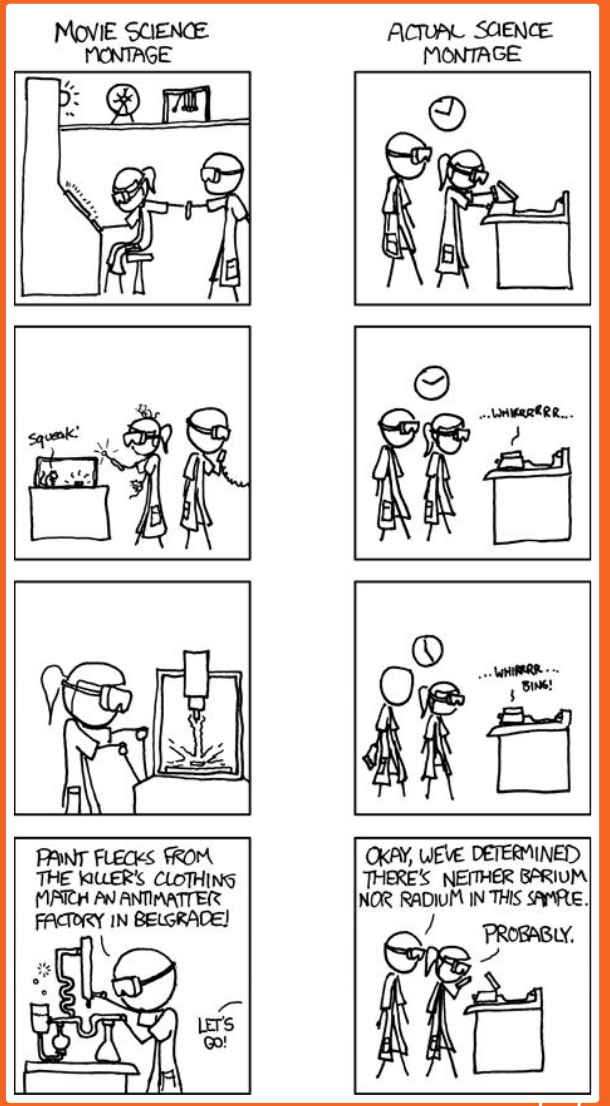
Foundations: **How to design experiments in NLU**



xkcd

Sam Bowman

NYU DS-GA 1012

(With pieces from Chris Potts and Bill MacCartney's Stanford CS 224U)

Thanks for the feedback so far!

Survey still open until late tonight—common themes so far:

Doable:

- Go over solutions to labs (and shorten the lab exercises a bit).
- Add documentation to the homeworks.
- Turn the lights up.

Not doable (sorry!):

- Emphasize more of the math.
- Change the participation rules.

Poll:
Last week.

Today


The California Agriculturist
AND
LIVE STOCK JOURNAL.
SAN JOSE, CAL., FEBRUARY, 1876

- If you want to publish an influential paper in NLP, the engineering is only half the work.
- This is the first of two *foundations* lectures about the other half:
  - How to find and understand related work on your problem
  - How to design effective experiments and analyze their results
  - How to stay out of ethical trouble
  - How to write and publish your work

Today


The California Agriculturist
AND
LIVE STOCK JOURNAL.
SAN JOSE, CAL., FEBRUARY, 1876

- If you want to publish an influential paper in NLP, the engineering is only half the work.
- This is the first of two *foundations* lectures about the other half:
  - **How to find and understand related work on your problem**
  - **How to design effective experiments and analyze their results**
  - How to stay out of ethical trouble
  - How to write and publish your work

Today


The California Agriculturist
AND
LIVE STOCK JOURNAL.
SAN JOSE, CAL., FEBRUARY, 1876
Vol. 7.—No. 2.

- This is not specific to the class project, but almost all of it applies.
- For a successful project, you must:
  - Identify an open problem, present a hypothesis about it, and survey the relevant literature.
  - Design and run an experiment to test that hypothesis.
  - Analyze the results to reveal what your experiment tells us about your hypothesis.

Reading Related Work

The literature review

- Do this early!
- Why?
  - Make sure that what you're doing hasn't already been done before.
    - If it has, and the paper is easy to find, you won't get full credit.
  - Learn about common methods, datasets, and libraries that will make your life easier.
  - Buy yourself more time to think about the questions that *haven't* been answered in the literature.

Identifying papers

1. Do a keyword search on Google Scholar (or the ACL Anthology)
2. Download the papers that seem most relevant
3. Skim the abstracts, intros, & previous work sections
4. Identify papers that look relevant, appear often, & have lots of citations on Google Scholar
5. Download those papers 6.
6. Return to step 3

Credit: Bill MacCartney

Anatomy of an NLP paper

Eight two-column pages plus 1-2 pages for references. Here are the typical components (section lengths will vary):

Title info 1. Intro	2. Prior lit.	3. Data	4. Your model
4. Your model	5. Results	6. Analysis	7. Conclusion

- Your class project gets only four pages, but it only needs to contain one experiment.

The literature review

- Where to find the most trustworthy papers:
  - NLP and Computational Linguistics: Proceedings of ACL conferences (ACL, NAACL, EACL, EMNLP, CoNLL), TACL, *Computational Linguistics*, arXiv*
  - Machine Learning/AI: Proceedings of NIPS, ICML, ICLR, AAAI, IJCAI, and arXiv*

* Official (reasonable) ACL policy: arXiv papers *are* prior work and should be in literature reviews and comparison tables, but you can still treat claims from those papers more skeptically.

The literature review

- You'll have to use your judgment in deciding which papers to read, and which papers to trust.
  - Papers can disagree with one another.
  - You might just find too many papers.

The literature review

- Which papers will be most useful?
  - Newer ones, especially if they cite the older papers that you're interested in.
    - The newer paper might contain a good summary of the older one!
  - Papers published in top conferences and journals, rather than arXiv papers or papers published elsewhere.
    - Reviewers have carefully looked at these papers for mistakes or inconsistencies.
  - *Published* papers with negative results (method X *doesn't work*, method X *doesn't* do what you think it does, ...), rather than papers with positive results.
    - Negative results are usually held to a higher standard in for publishing.

Organizing a Project & Doing Good Science

The Hypothesis

- Bad/unfalsifiable:
  - *Neural networks are more elegant and principled than feature-based systems for sentiment analysis.*
- Falsifiable but uninformative:
  - *My convolutional neural network (CNN) model outperforms the Socher et al. (2015) baseline on the Stanford Sentiment Treebank data.*
- Typical good paper:
  - *CNN models outperform feature-based systems on sentiment analysis for reviews.*
- Ambitious paper:
  - *Non-professional web users tend to express sentiment using a set of common* fixed phrases *when they discuss products. These phrases vary enough that simple symbolic features don't capture them well, but they're structured enough that filter-based neural networks like CNNs can capture them very efficiently.*

Controlled experiments

Your experiment needs to test your hypothesis. Everything that you've learned about *controlled* experimentation applies in NLP.

Example results on SNLI:

- Paper A: LSTM gets 77.6%
- Paper B: CBOW gets 80.6%

What can you conclude?

Controlled experiments

Your experiment needs to test your hypothesis. Everything that you've learned about *controlled* experimentation applies in NLP.

Example results on SNLI:

- Paper A: LSTM gets 77.6%
- Paper B: CBOW gets 80.6%
- Paper C: GRU gets 80.7%

What can you conclude?

Controlled experiments

Your experiment needs to test your hypothesis. Everything that you've learned about *controlled* experimentation applies in NLP.

Example results on SNLI:

- Paper A: LSTM gets 77.6%
- Paper B: CBOW gets 80.6%
- Paper C: GRU gets 80.7%
- Paper C: BiLSTM gets 84.5%
  - With the same input preprocessing, hyperparameter tuning method, etc.

What can you conclude?

Baselines

- *Baseline* can mean any of three things, and you usually want all three:
  - Anyone's performance numbers for *simplest reasonable approach* to your problem (CBOW, logistic regression, plan seq2seq).
  - *Your* numbers for a reasonably competitive system based on existing ideas. This can be based on a published system or your own work, but you should make a precise, controlled comparison with it.
  - The *best published number* for your problem. It's common to not beat this, but you should compare with it. This is what could give you the right to claim the 'state of the art'.

Upper bounds

- Less necessary, but often helpful: Measure (or find out) the accuracy of a human annotator!
  - You can't use the same annotations that you used to create the data.
  - It's better if you don't even use the same people. (Why?)
- 77% accuracy doesn't look so bad if a human only gets 81%!

Datasets

- You'll want real natural language data for any NLP project. For a publication, you'll usually want multiple datasets.
- You'll almost always want to use at least one dataset that appeared in related prior work, even if you're the first one to solve a specific problem.
  - Working on sentiment analysis in Portuguese?
    - Show that your method is competitive on English datasets like SST or IMDB.
    - Collect or create a dataset for Portuguese.
    - Show results on the new data with your method and several existing baselines.

Getting datasets

- Find them!
  - The ACL anthology!
  - The Linguistic Data Consortium (free membership through NYU)!
- Build them!
  - Write *very* detailed notes on what you do and why: Readers and reviewers will assume that all of your decisions are biased to that unfairly favors the systems you're interested in. To convince them that you didn't, you'll need to show that you made fair and reasonable decisions.
- Scrape them?
  - It's easy to break the law this way. Or get NYU temporarily banned from Twitter/eBay/Reddit/Tinder/etc.
  - For some slightly dated discussion of scraping, see: http://nlp.stanford.edu/IR-book/

Getting datasets

- Have someone else build them!
  - Write *simple* annotation guidelines that non-NLPers can follow.
    - You still need to convince your reviewers these guidelines aren't unfairly advantaging your method.
  - Pay some friends or Mechanical Turk (MTurk) workers to follow your guidelines.
  - This is often quick and cheap in practice. Ballpark costs (at $10–20/hr):
    - Writing: ~$0.05–$0.25 per sentence written
    - Labeling: ~$0.02–$0.15 per sentence read
  - Some advice:
    - Try your annotation task yourself for half an hour.
    - Communicate with your annotators! Even on MTurk, they'll often be trying hard to understand your task, and you want to be available if they have questions or if they see issues.

Getting datasets

- Have someone else build them!
  - Some advice:
    - Try your annotation task yourself for half an hour. If there's anything that you find confusing or frustrating, fix it!
    - Communicate with your annotators! Even on MTurk, they'll often be trying hard to understand your task, and you want to be available if they have questions or if they see issues.
  - For this class:
    - You're welcome to use MTurk *if* you have access to research funding (mostly PhD students).
    - Otherwise, consider trading data with another team, and having the members of the other team annotate your data.
    - Don't tell them too much about your project—just share the annotation guidelines.

Artificial datasets

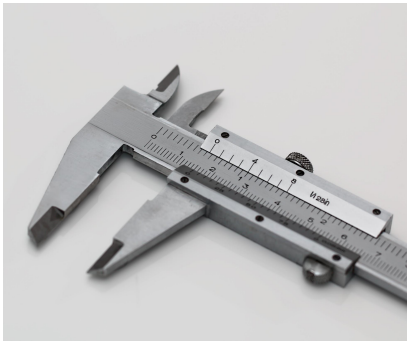
- Artificial (machine-generated) datasets can allow you to run preliminary experiments in minutes, minimizing the risk that you'll spend hours or days running something that won't work.
- They can also help you with error analysis, by revealing what kinds of pattern your system can/can't learn.
- Artificial datasets ***won't*** convince NLP readers that your system works on language.
  - Case study BaBi QA:

```
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary?          bathroom
```


Intermission

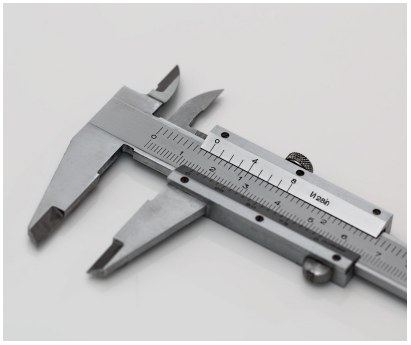
Organizing a Project & Doing Good Science

Quantitative evaluation



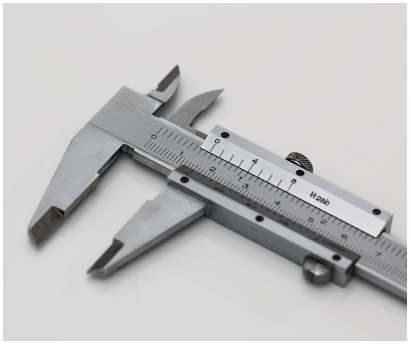
- This is a huge topic, but in short:
  - Follow prior work ***precisely*** in how you choose and implement your main evaluation metric.
  - *Do* show metrics as many variants of your model as you can (ablation analysis).
    - Example:
      - **CNN sentiment classifier with GloVe embeddings**
      - CNN sentiment classifier with random embeddings
      - LSTM sentiment classifier with GloVe embeddings
      - CBOW classifier with GloVe embeddings
      - CBOW classifier with random embeddings
    - Should you use the dev set or the test set?
  - *Do* use carefully-designed human evaluations for tasks where this is standard (dialog, summarization, etc.).

Quantitative evaluation



- This is a huge topic, but in short:
  - *Do* invent new analysis metrics if they help you make your point.
    - If your system good at classifying long sentences, also report accuracy on the subset of sentences of length >20.
    - If your baseline for a text generation task only uses very common words, and your system fixes that, measure the *average frequency* of the words that each system generates.
    - *Don't* talk about the 'state of the art' for things that nobody else measures.
  - *Do* perform extrinsic evaluations on downstream tasks if you expect the output of your system to be used as the input to any other system.

Quantitative evaluation



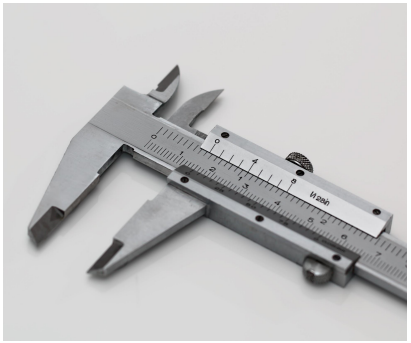
- This is a huge topic, but in short:
  - *Do* explicitly test for statistical significance, especially when your hypothesis depends on a small difference or when model performance is highly variable. (See Resnik and Lin reading or Berg-Kirkpatrick et al. '12)
    - Methods here vary widely within NLP, but don't expect readers to take you seriously if the main claim of your paper is that you get an 0.2% accuracy improvement on the state of the art.

Quantitative Evaluation

- If your system doesn't beat your baselines, or if the differences between your results aren't significant: Say so! And explain what you found!
  - Well-presented negative results do move the field forward, and are in common in projects like class papers that have to happen in a short time.

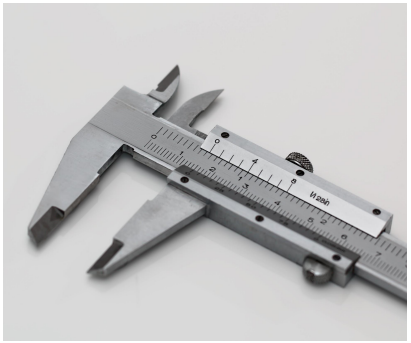
(barely) not statistically significant (p=0.052)
a borderline significant trend (p=0.09)
a certain trend toward significance (p=0.08)
a clear tendency to significance (p=0.052)
a clear, strong trend (p=0.09)
a decreasing trend (p=0.09)
a definite trend (p=0.08)
a distinct trend toward significance (p=0.07)
a favorable trend (p=0.09)
a favourable statistical trend (p=0.09)
a little significant (p<0.1)
a margin at the edge of significance (p=0.0608)
a marginal trend (p=0.09)
a marginal trend toward significance (p=0.052)
a marked trend (p=0.07)
a mild trend (p<0.09)
a near-significant trend (p=0.07)
a nonsignificant trend (p<0.1)
a notable trend (p<0.1)
a numerical increasing trend (p=0.09)
a numerical trend (p=0.09)
a positive trend (p=0.09)
a possible trend toward significance (p=0.052)
a pronounced trend (p=0.09)
a reliable trend (p=0.058)
a robust trend toward significance (p=0.0503)
a significant trend (p=0.09)
just lacked significance (p=0.053)
just marginally significant (p=0.0562)
just missing significance (p=0.07)
just on the verge of significance (p=0.06)
just outside levels of significance (p<0.08)
just outside the bounds of significance (p=0.06)
just outside the level of significance (p=0.0683)
just outside the limits of significance (p=0.06)
just short of significance (p=0.07)
just shy of significance (p=0.053)
just tendentially significant (p=0.056)
leaning towards significance (p=0.15)
leaning towards statistical significance (p=0.06)
likely to be significant (p=0.054)
loosely significant (p=0.10)
marginal significance (p=0.07)
marginally and negatively significant (p=0.08)
marginally insignificant (p=0.08)
marginally nonsignificant (p=0.096)
marginally outside the level of significance
marginally significant (p>=0.1)
marginally significant tendency (p=0.08)
marginally statistically significant (p=0.08)
may not be significant (p=0.06)
medium level of significance (p=0.051)
mildly significant (p=0.07)
moderately significant (p>0.11)
slightly significant (p=0.09)
somewhat marginally significant (p>0.055)
somewhat short of significance (p=0.07)
somewhat significant (p=0.23)
strong trend toward significance (p=0.08)
sufficiently close to significance (p=0.07)
suggestive of a significant trend (p=0.08)
suggestive of statistical significance (p=0.06)
suggestively significant (p=0.064)
tantalisingly close to significance (p=0.104)
technically not significant (p=0.06)
teetering on the brink of significance (p=0.06)
tended toward significance (p=0.13)
tentatively significant (p=0.107)
trend in a significant direction (p=0.09)
trending towards significant (p=0.099)
vaguely significant (p>0.2)
verging on significance (p=0.056)
very narrowly missed significance (p<0.06)
very nearly significant (p=0.0656)
very slightly non-significant (p=0.10)
very slightly significant (p<0.1)
virtually significant (p=0.059)
weak significance (p>0.10)
weakly significant (p=0.11)
weakly statistically significant (p=0.0557)
well-nigh significant (p=0.11)

Qualitative evaluation and error analysis



- NLP papers generally have an *analysis* section.
  - This may be called something else!
- Your goal here: Convince the reader of your hypothesis.
- If your hypothesis is interesting, it'll be hard to evaluate with standard/intuitive quantitative metrics.
  - *Non-professional web users tend to express sentiment using a set of common fixed phrases when they discuss products. These phrases vary enough that simple symbolic features don't capture them well, but they're structured enough that filter-based neural networks like CNNs can capture them very efficiently.*

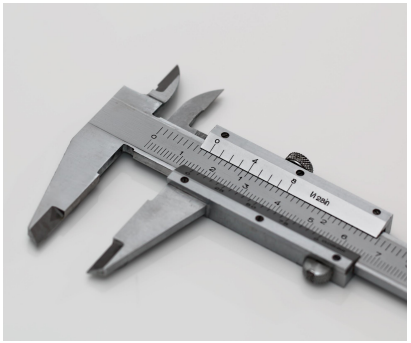
Qualitative evaluation and error analysis



- So, give the reader qualitative evidence, too!
- Places to start:
  - Look to prior work, and do what they do.
  - Show examples of system output.
  - Come up with *categories* to describe system errors and count them.
  - Visualize your hidden states with tools like LSTMVis.
  - Plot how your model performance varies with amount of data.
  - Build an online demo!

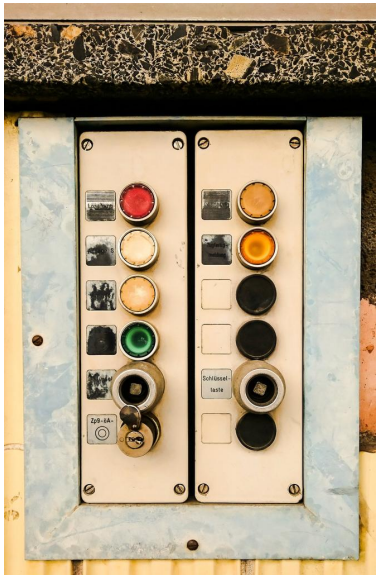
Formative vs. summative evaluation

When the cook tastes the soup, that's formative; when the customer tastes the soup, that's summative.



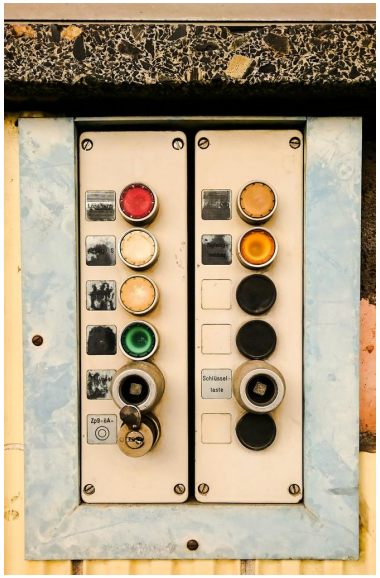
- Formative evaluation: guiding further investigations
  - Typically: lightweight, automatic, intrinsic
  - Compare design option A to option B
  - Tune hyperparameters: smoothing, weighting, learning rate
- Summative evaluation: reporting results
  - Compare your approach to previous approaches
  - Compare different *major* variants of your approach
  - Only use the test set here
  - Generally only bother with human or extrinsic evaluations here
- Potential serious mistake: Don't save all your qualitative evaluation for the summative evaluation!

Hyperparameter tuning



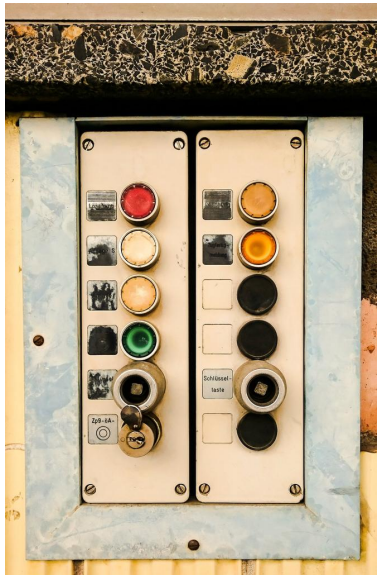
- **Crucial point:**
  - You must tune the hyperparameters of your baselines just as thoroughly as you tune them for any new model you propose!
  - Failure to do this invalidates your comparisons, and depending on how you write the paper, could border on academic misconduct. (Related reminder: Don't tune on the test set!)
- Read the fine print while you're doing your literature review to get a sense of what hyperparameters to worry about and what kinds of value to expect.
- If you're not sure whether to tune a hyperparameter, you probably should.

Hyperparameter tuning



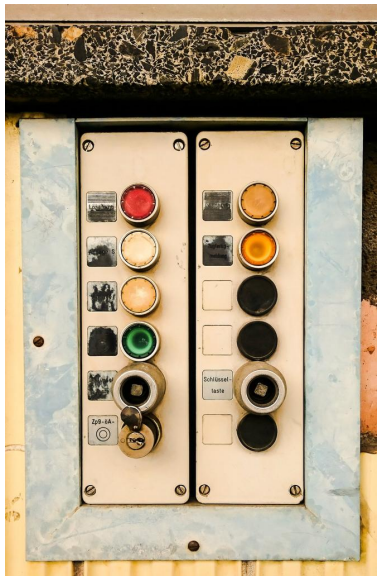
- Grid search: Inefficient.
- Bayesian optimization: Optimal, but public packages aren't great.
- Random search (Bergstra and Bengio '12): Easy, and near-optimal.

Hyperparameter tuning

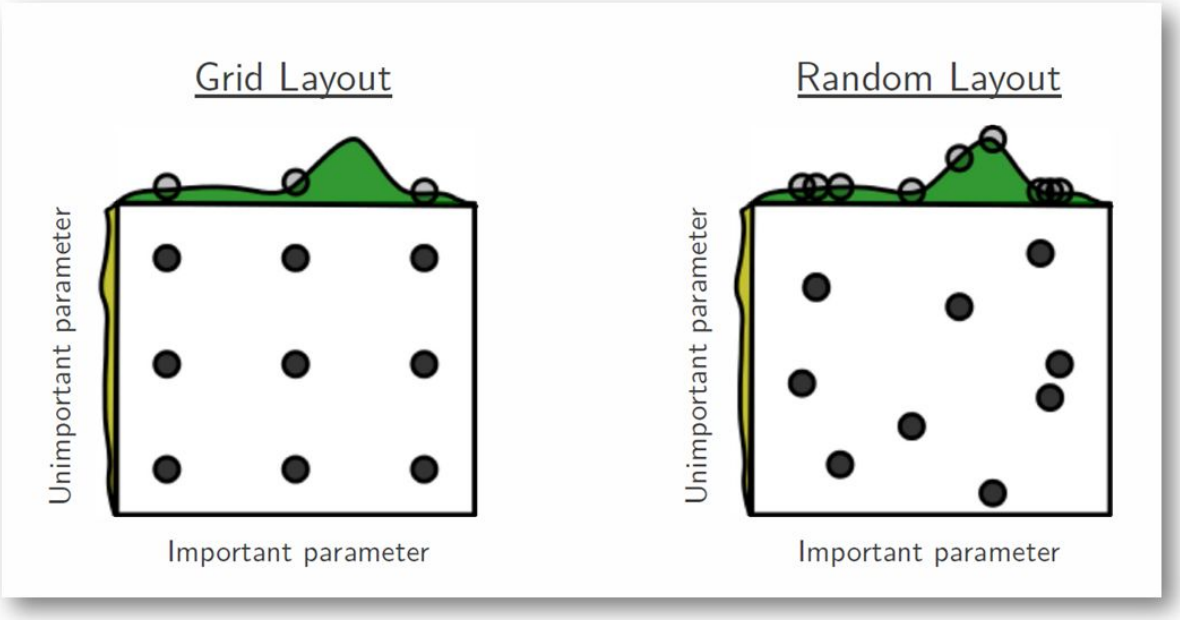


- Random search (Bergstra and Bengio ‘12):
  - Define distributions over all your hyperparameters.
  - Sample N times for N experiments.
  - Look for patterns in your results.
  - Adjust the distributions and repeat until you run out of resources or performance stops improving.

Hyperparameter tuning



- Random search (Bergstra and Bengio ‘12):
- Why?



Iterative development

- As soon as you start:
  - Create a git repo for your project.
  - Find or build code to load your data.
  - Find (try not to build) code to evaluate results.
  - Find or build a *very simple* baseline.
- After that, keep track of:
  - Commands and git checkpoint identifiers for each of your experiments.
  - Saved model checkpoint files for all reasonably effective/interesting experiments.
  - Notes on what each experiment was meant to test.
- Done this way, research is an *anytime algorithm*.
  - All of these are good practices for independent reasons, and they ensure that you'll be able to write a paper on time even if your more ambitious plans don't work out.

Misc.

Recommended reading

- Resnik and Lin on evaluation
- Gardner slides on engineering
- Jason Eisner's advice posts on NLP/ML research

Typesetting and LaTeX

- You'll need to use the *LaTeX* typesetting tool to write your paper.
- For the basics of LaTeX, there are good tutorials online.
- Tomorrow's lab will cover a more advanced tips that are useful for writing NLP/ML/AI papers.
- Another good source of handy tricks:

Coming up

- Next week:
  - Nothing! Spring break!
- March 27:
  - Syntax and parsing
  - HW3 out on dependency parsing
- April 3:
  - Part 2 of *the other half*: Writing, ethics, and publishing
  - HW3 due

...

- May 1:
  - Partial draft paper due