Hyperreal Enterprises: Roadmap

Joe Corneli

November 3, 2020

Abstract

This document can be thought of as an informal outline of a "*tactic state*". On a technical level, the upstream source for this material is an Org Roam graph. The "Wiki" section contains instructions for accessing the material and generating derived formats, such as the Org Agenda.

Contents

1 Hyperreal Enterprises: Roadmap

1.1 Preface

This document synthesises a Roadmap (and perhaps also a Product Roadmap) for Hyperreal Enterprises, Ltd. The Roadmap is being written inside Org Roam (an Emacs package), and shared via Git on repo.or.cz. This document can be thought of and edited as a Wiki, from which other downstream formats can be derived. A narrative introduction follows. To skip it, browse ahead to Top.

1.2 Introduction

1.2.1 Applying AI to technical fields is a huge opportunity.

It's striking that in computer programming work, *collective intelligence* is used almost everywhere, but artificial intelligence is used almost nowhere. AI can kick butt at Chess and Go, but university courses are still taught by human professors. Surely, AI that could write code at a human level would be a valuable thing...

1.2.2 But where's the human-level AI for coding?

A research agenda around "AI for programming" was already spelled out by Alan Turing in the late 1940s. It is only relatively recently that we have massive amounts of relevant data to work with. Code completion and code generation tools are some of the "low hanging fruit" of this domain. We think that something much more substantial is around the corner.

1.2.3 We are looking at this problem as an informal open source R&D collective.

Our team brings together applied experience in Natural Language Processing (NLP), computational modelling, online communities, physics, biology, mathematics, and statistics. We've been getting together for daily coffee chats, and sharing information and skills with each other in online seminars. Sometimes we bring in guest speakers. We hope as time goes by our chats will coalesce into notes, blog posts, papers, and working prototypes. We aren't promising a schedule of deliverables — because we're all volunteers — and we're doing this for fun and interest. But we do plan to share what we're learning as we go along!

1.2.4 Broadly, the steps we have in mind go from data, to models, to AI agents.

We plan to document our progress (or lack thereof) on this blog. As a very rough outline, this is what we expect to look at:

- We plan to use contemporary information extraction methods to derive computationally meaningful material from Stack Exchange Q&A, Github Issues, and programmers' discussions, along with code. To do this, we will combine general purpose language models, like BERT, and a knowledge graph approach.
- We plan to use category theoretic methods as a glue that can hold together a range of computational models, including models of programs and the process of computer programming. Monocl is an existing process modelling language that has been used to create an abstraction layer over a collection of data science programs. We plan to generalise this.
- Ultimately, we plan to install these models in computational agents who can then "converse with each other to sharpen their wits," as Turing anticipated, mirroring contemporary developments in self-play. However, the specific design of the agents remains an open issue at the moment! We plan to explore techniques from Bayesian learning, logic programming, and reinforcement learning.

1.2.5 Reflection is part of the process.

We're interested in understanding human behaviour as well as technical topics: that goes for our own behaviour in particular. We plan to post to our blog at least monthly — as long as we keep our discussion running — using it as a place to reflect on how things are going. Although we're grappling with some weighty topics, success is not all or nothing! Writing here will mark our progress and be useful in their own right (e.g., blog posts can feed into research papers or tools).

1.2.6 Progress so far

Alongside setting up a blog and drafting an initial anouncement (voilà!), we've updated the website of our affiliated UK-based company Hyperreal Enterprises to match the outlines of what we've described here. As mentioned above, are having regular meetings, which we'll record if they look likely to be interesting to a wider audience. We've got nice internal documentation going, via Org Roam, from wich various derived formats are produced, including a public wiki generated by Firn. (We recently sent an abstract to EmacsConf 2020 to talk about the various other Emacs-based tools we're using!)

Some of our seminars so far have explored an Open Information Extraction (with several demo notebooks), and looked at the early parts of Michael Betancourt's "Towards a principled Bayesian workflow".

Active channels on our Discord server include:

- #emacs-cloud-hypernotebooks
- #how-to-design-programs
- #text-analysis
- #model-construction

- #knowledge-graph

1.2.7 TL;DR

We are creating AI tools that will process open source information and build on these in applications such as automated programming.

2 Top

2.1 Motivation: For the sake of advancing AI

We are doing this R&D work partly to make demonstrations of more advanced AI systems. We expect that our stance on AI will not necessarily be a popular one. But this is an important "minor" strand of AI research dating back to Alan Turing:

> "As time goes on the [computer] itself will take over the functions both of [programmers] and of [users]...The [programmers] are liable to get replaced because as soon as any technique becomes at all stereotyped it becomes possible to devise a system of instruction tables which will enable the electronic computer to do it for itself. It may happen however that the [programmers] will refuse to do this. They may be unwilling to let their jobs be stolen from them in this way. In that case they would surround the whole of their work with mystery and make excuses, couched in well chosen gibberish, whenever any dangerous suggestions were made."
> – Alan Turing, 1947.

On average, advanced AI would bring in new ways of working, and would facilitate broad access to high-quality training. This agenda could serve to focus the mind of technical workers, but not many are pursuing it presently.

2.2 Motivation: Technical experiments become easier

Even in the present time, without relying on any speculative AI futures to magically appear, we can benefit from pursuing the agenda above. Accordingly, we are doing some applied work with existing software that will give us a set of further tools and levers to work with.

2.3 Representative Prior Work

2.3.1 PlanetMath

PlanetMath users created a reasonably large informal mathematical knowledge base together. On the way, we came up with several technical demos and sketched possible previews for upcoming features. One possible direction of work we looked at would be to focus on building a comprehensive category theory knowledge base.

2.3.2 Modelling the way mathematics is actually done

In this paper, we talked about how mathematics is situated somewhere in between 'games' and 'storytelling' in its complexity. We proposed to build computational models of informal mathematical reasoning. Subsequent work continued on in this direction, using ideas from dialogue games and argumentation theory more broadly.

2.4 A sketch of a plan

So, having gotten together around these ideas, we're having online chat, frequent short meetings. We've talked about maintaining a blog that would describe what we're learning and developing. So, roughly speaking, we will try to develop a curriculum through the blog. We also have this wiki, that any of us can edit, which we can use as a staging ground for more developed blog posts. Our thought was that blog posts might move in the direction of more developed outputs, whether products or research papers. We want to use some ideas adapted from Scrum to build a shared awareness of what's going on. However, we want to be careful not to become "managerial" since everyone is currently here as a volunteer, working on topics of his or her own interest. We want to provide mutual support and fun. Reflection, whether in writing, or by recording and listening again to conversations, should help with that. We are not constraining things to come out in a purely structured curriculum, or any other form of product development. "Users" and "customers" may appear as we release things we are happy with and expand our little community.

> "Rousseau says, someone who has been properly educated will be engaged in society, but relate to his or her fellow citizens in a natural way. ... We naturally look after our own preservation and interests. By contrast, *amour-propre* is an unnatural self-love that is essentially relational. ... Thus, *amour-propre* can contribute positively to human freedom and even virtue. Nevertheless, *amour-propre* is also extremely dangerous because it is so easily corruptible. ... In its corrupted form, *amour-propre* is the source of vice and misery, and results in human beings basing their own self worth on their feeling of superiority over others." — IEP

2.5 A possible formulation: short correlated sprints as opposed to random behaviour

"Two people working together 4 hours a week for two weeks" could serve as an approximate unit of work. Once we have amassed a few outputs from this kind of effort, we will have some evidence of the kinds of things that we can realistically achieve. So far, our workflow has been more based on solo activities and informal conversations, but short robust team-ups continue to be an option!

> Hypothetical conversation: *In my next post I want to integrate something that I learned from you about PL. I want to drive in the direction of synthesis, as hard as I know how to right now. This depends on everyone having free time to invest in this. Start a blog where we think about what's the overlap in terms of learning?*

3 Why not what

Our purpose:

- We want to make the knowledge economy accessible to everyone.
- Our long-term vision is computational intelligence based on collective intelligence.

3.1 Teach arbitrary coding

This would be an abstraction over teaching basic programming and knowledge graphs.

3.1.1 Feature: Production system

We've started to build a simple production system that can be used to detect errors in subtraction (reimplementing some classic work). We were thinking that something similar could be used to detect other kinds of errors (so, for debugging, teaching), and to support other kinds of reasoning processes (e.g., turning Q's into A's in a question-answering system).

We previously did a little exploratory work, with similar intent, using polygraphs as input, in the workshop paper *Modelling the Way Mathematics Is Actually Done*.

1. Demo application: Reimplementing classic rules to model subtraction

   We looked at a classic paper about "subtraction on Mars" and it seems that reimplementing it might be the best way to go.

3.2 How to Design Programs

We were thinking of *How to Design Programs* (HtDP) as a potential basis for this work. We would want to respect category theoretic concepts in the presentation. We would expect to find analogues in settings like Bayesian modelling.

We could proceed by looking at relationships with argumentation theory, thinking about how to do this in a theoretically consistent way. Once we have a definition of the programming language we're going to use, we can then do argumentation over that.

Another strategy would be to develop a DSL for HtDP ideas, which we could then reuseq to generate patterns for learning how to design various structures (say, web pages or probabilistic programs). To do this well you'd need ways to express 'recipes'. For example, an MVP might be based on representing HtDP-style recipes using sequent calculi for session types. These represent interactive protocols.

You'd use cut-elimination to have two players interact (using something like the Lakatos Game diagram). But what formalism would you use? E.g., *geometry of interaction in linear logic* has been used for this kind of thing, but could it be used here? With a suitable formalism in place we would then imagine that a computer programming agent would just follow the "Lakatos Game" style HtDP script. So, this would contribute to the development of agent models for programming and program-related Q&A.

3.2.1 Related work

- General theory-informed algorithms (e.g., apply category theory to scientific models).
- K framework: Have transformations for any language you define in it.
- HtDP is similar applied to programming teaching. Start with PL theory and then find universal things.
- How can we define statistics in a general way and then derive things from it? (E.g., Anglican probabilistic programming?)

4 Construct, critique, improve models of the creative process

We want tools and processes for working with models, with a particular emphasis on improved models of the creative process. The reason for this emphasis is that if we have good models of the creative process, including the modelling process, we can then apply them to a wide range of problems! This prompts reflection on the infrastructure and tools that we are actually using.

4.1 Emacs Hyper Notebook

We are developing a better way to do "Jupyter notebooks" using Emacs. This recovers some of the Research Collab ideas developed by Aaron Krowne. It should integrate features such as writing and task management (e.g., *Org*) Program evaluation (e.g., *Maxima*), Typesetting and presentation (e.g., slides via $\LaTeX$), and navigation (e.g., *Org Roam* for displaying topics as a graph). We should be clear that the various technologies used are slot-fillers and they might be replaced with other things, or augmented (e.g., *Lean* for formal verification of some of the above?). A useful input to this process would be implementation of examples without integration. This can then be redone in a more integrated fashion.

An integration using existing technologies will have limitations, once we have this demos then we will see some of the gaps and how more advanced tech could be useful. (For example, Ray's work with Gerschom could turn out to be useful here.)

1. Some quick thoughts

    - If it was sitting inside a web container, then maybe it's a quickstart thing that comes in a user friendly form.
    - Maybe add menu-bar items to make it look like a web browser.
    - Emacs Maxima interface, we might build on it — for Monday 12th October, a quick "15 minutes" talk to catch up
    - ... possible deliverable for later on: a NIST talk?

4.1.1 TODO Figure out subtasks to deliver for EmacsConf JOE:RAY:CAMERON

4.1.2 TODO Figure out how EHN relates to other projects JOE:RAY:CAMERON

4.1.3 TODO Keep testing crdt.el and lockstep.el JOE:RAY

4.1.4 TODO Could Emacsconf talk become a blog post? JOE:RAY:CAMERON

(Notice that with crdt, typing can go on inside folded nodes! Qiantan is thinking about a mode to make overlays shareable, which would change things a bit.)

4.1.5 Partial prototypes

How far can we go... Through Roam? (We could at least talk to Connor about Roam on Twitter?) Through Jupyter? Foam? Glamorous Toolkit? Can we integrate what we're building with existing tools like these? Do Lenses or other kinds of ACT machinery help with this at all? Would our system potentially play a role as a universal backend?

4.1.6 Feature: Arxana 2020

Revisit Arxana and turn it into something that we can actually use. This is rather closely related to the use of "knowledge graph" formulations we've been discussing, since Arxana allows us to combine writing with knowledge representations. In our last round of work with Arxana, we left off at the point of integrating logic programming into the system.

4.1.7 Links to useful resources

Technology like this could be used to build simple demos (e.g., Emacs in the browser, running Org Mode). We've noticed some other related tools as well, like Organice and CodiOrg that could provide alternative interfaces.

- exp2exp/notebooks: This is a Docker configuration for running jupyter with multiple kernels on Arch Linux.
- emacsclient Options - GNU Emacs Manual
- butlerx/wetty: Terminal in browser over http/https. (Ajaxterm/Anyterm alternative, but much better)
- xtermjs/xterm.js: A terminal for the web
- https://twitter.com/cianbutlerx
- tsl0922/ttyd: Share your terminal over the web
- yudai/gotty: Share your terminal as a web application
- butlerx/wetty - Docker Hub

- Setup Web Terminal using Wetty Docker Image | by Chairat Onyaem (Par) | Medium
- krishnasrinivas/wetty - Docker Hub

4.1.8 Other related work

- James Fairbanks (relate this to Betancourt).

4.1.9 Testing

```
(def a 2)
```

```
#'user/a
```

```
a
```

```
2
```

```
(range 10)
(def a 1)
```

```
| (0 1 2 3 4 5 6 7 8 9) |
| #'user/a              |
```

```
a
```

```
1
```

4.1.10 Implementing a quick demo for Emacs NYC

Notes for the first talk will appear here:

- Hypernotebook First Demo

4.1.11 What would we actually want as our org interface?

We had a short problem with this:

```
1+1;
(error "No such language mode: nil-mode")
...
```

4.1.12 Backends

1. jupyter The jupyter backend works well locally but suffers from a bug when run via tramp. See emacs-jupyter remote debugging
2. ob-streams

   This is work in progress, with some sample content above.

4.1.13 Future work

- Extending to VS Code? Would people who use VS code even want this kind of interaction? Maybe VS Code is better for quick visualisations?

4.2 Hypernotebook First Demo

First demo notes will go here.

4.2.1 During our first session on this, we set up a calculator

We wrote up a configuration, starting from what the `src` block might look like:

- `:process` stands in for `:session` now as an alternative
- `org-babel-execute-src-block` this is the function that will be called
- We saw we had to hang into the `lang` parameter of the above function, but override using `:process`
- The variable `org-stream-output` needs to be defined (it's in our `ob-stream.el`).
- The function `org-babel-stream` needs to be defined as well (it's in our `ob-stream.el`).
- If you're going to use a language mode like `calc` it should be required.
- The process itself needs to be defined as well.
- `(setq org-confirm-babel-evaluate nil)` should be set.

  This is an example of what we came up with:
  `#+begin_src calc :stream calculator :results output org`

```
10*8
```

```
80
```

4.2.2 Background research

We took a good look at emacs-jupyter to understand how it works. It seems to work reasonably well but not able to do all the hops via TRAMP to connect with a notebook running on Google Cloud. This is documented as bug #191 in the emacs-jupyter repo.

4.2.3 Now we wanted to improve this to make it more robust

- Be able to handle multiple backends (now via "servants" for different shell commands and potentially other processes), not just the `bc` calculator!
- In the first version we just used a variable to store things, now use a hash table to organise the data better (`org-babel-servant-info`)
- Generalise everything

4.2.4 Stitching things together

- Find a good way to weave `org-babel-servant` into `org-babel-execute-src-block`
- Organise the callbacks, can we demo it calling Maxima?

1. Getting things going

```
;; Here we start the process
(get-buffer-create "maxima-error")
(setq maxima-proc
      (make-process
       :name "maxima-proc"
     :command '("maxima" "--very-quiet") ;; if we need parameters can add here
       :stderr "maxima-error"
       :filter #'org-stream-string-callback))

(setq calculator-proc
      (make-process
       :name "calculator-proc"
       :command '("bc" "-q") ;; if we need parameters can add here
       ;; :stderr "maxima-error"
       :filter #'org-stream-number-callback))
```

(a) Calculator example again

```
10+1
```

```
10
```

(b) Maxima example

Here we call the process we just started.

```
3+600000000;
```

```
600000003
```

```
display2d:false;
```

```
false
```

```
expand((x+1)^9)
```

```
$$x^9+9\,x^8+36\,x^7+84\,x^6+126\,x^5+126\,x^4+84\,x^3+36\,x^2+9\,x+1$$
```

```
expand((x+1)^1)
```

```
$$x+1$$
```

4.2.5 Ob-servant example

1. Calculator example again

```
10+1
```

```
11
```

4.2.6 Next steps

- Change the formatting of the output so it doesn't come across as a table
- Carry on with ob-servant to integrate it (improve calling Maxima, errors)
- Raise the change with the Org maintainers
- Package it up! (patch + repo?)
- 5 minute talk for Emacs New York (Monday 2 November 2020).

4.2.7 Social Networking

- Talk with Fermin to understand what he's doing with Maxima and see if we need to do anything differently

4.2.8 Final polishing

- Consider renaming it to ob-servent!

4.3 emacs-jupyter remote debugging

4.3.1 Debugging

1. Initial fix (of what turned out to be a minor problem)
    - `jupyter-start-kernel` was problematic, because the id didn't seem to be set correctly
        - line -2 from end of this function, added `:id (substring conn-file -41 -5)`.
2. Ongoing concerns: can we access the kernel?
    - However, this still doesn't solve our problem
        - Now we are debugging `jupyter-kernel-info`
        - Our aim is to figure out `jupyter-send-kernel-info-request`, maybe also `jupyter-wait-until-received`
    - jupyter-comm-initialize
    - jupyter-kernel-info
    - jupyter-make-client
    - jupyter-session —
    - jupyter-kernel-info — doesn't get a message back for some reason

    We suspect it looks at the wrong kernel b/c it examines an id that doesn't seem to exist on the server.

    ```
    (with-slots (kernel) manager
      (oref kernel session))
    ```

4.3.2 Set up container on gcp

1. gcp configuration

    You may like to run `gcloud auth login` (auth login docs). This is an interactive process that launches oauth for your google account in the web browser so I think it is best to do it from a terminal though it may be possible to run it in org-babel.

    ```
    gcloud config configurations list
    ```

    ```
    NAME    IS_ACTIVE  ACCOUNT              PROJECT  COMPUTE_DEFAULT_ZONE  COMPUTE_DEFAULT_REGION
    default  True       holtzermann17@gmail.com  quarere
    ```

    ```
    gcloud config configurations describe quarere
    ```

    ```
    [1;31mERROR:[0m (gcloud.config.configurations.describe) The configuration [quarere] does n
    ```

2. launch container image

   Deploy a vm based on the container `cameronraysmith/notebooks:latest`.

   ```
   gcloud compute instances create-with-container notebooks-vm \
     --container-image registry.hub.docker.com/cameronraysmith/notebooks:latest \
       --container-restart-policy on-failure \
       --container-privileged \
       --container-stdin \
       --container-tty \
     --container-mount-host-path mount-path=/home/jupyter,host-path=/tmp,mode=rw \
       --machine-type n1-standard-1 \
       --boot-disk-size 50GB \
       --preemptible
   ```

   Setup ssh with your new instance

   ```
   gcloud compute config-ssh
   cat ~/.ssh/config | grep "Host notebooks"
   ```

   ```
   You should now be able to use ssh/scp with your instances.
   For example, try running:
   ssh notebooks-vm.us-central1-f.quarere
   Host notebooks-vm.us-central1-f.quarere
   ```

   You can `ssh` into the host machine or the container using the various commands below.

   ```
   gcloud compute ssh notebooks-vm # into host machine
   ssh notebooks-vm.us-central1-f.quarere docker ps -aqf "name=klt-notebooks-
   vm-cjme" # check the container ID
   gcloud compute ssh notebooks-vm --container klt-notebooks-vm-cjme # use gcloud ssh with -
   -dry-run to print the command
   ssh -t notebooks-vm.us-central1-f.quarere -- sudo docker exec -it klt-notebooks-
   vm-cjme /bin/sh # this takes you directly into the container
   ```

   Of course you can stop and start the machine with

   ```
   gcloud compute instances stop notebooks-vm
   gcloud compute instances start notebooks-vm
   ```

4.3.3 Startup the cloud vm running our container of interest

1. Setup remote container host machine

   We already setup the container named `notebooks-vm` so all we need to do to begin with is to start it up.

   ```
   gcloud compute instances start notebooks-vm
   ```

   Check that our instance is indeed running

   ```
   gcloud compute instances list
   ```

```
NAME         ZONE           MACHINE_TYPE  PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP    STATUS
notebooks    us-central1-c  n1-standard-1              10.128.0.22                 TERMINATED
notebooks-vm us-central1-f  n1-standard-1 true         10.128.0.26  104.154.99.181 RUNNING
```

Make sure the correct ip address is entered into our `.ssh/config` file.

```
gcloud compute config-ssh
```

```
You should now be able to use ssh/scp with your instances.
For example, try running:

  $ ssh notebooks-vm.us-central1-f.quarere
```

Inspect the IP address we find in our `.ssh/config` file

```
grep HostName ~/.ssh/config
```

```
HostName 104.154.99.181
```

(a) Execute commands on the remote container host machine

```
hostname --long
```

```
$ notebooks-vm.us-central1-f.c.quarere.internal
```

```
docker container ls
```

```
$ CONTAINER ID      IMAGE                                                COMMAND
4f31864fade2      registry.hub.docker.com/cameronraysmith/notebooks:latest         "/
c 'jupyter…"   23 minutes ago      Up 23 minutes                        klt-
notebooks-vm-cjme
19a4598c7503             gcr.io/stackdriver-agents/stackdriver-logging-
agent:0.2-1.5.33-1-1   "/entrypoint.sh /usr…"   24 minutes ago     Up 24 minutes
logging-agent
```

```
docker container ls
```

```
$ CONTAINER ID      IMAGE                          COMMAND              CREATED          STATUS
caadc9a126bb            gcr.io/inverting-proxy/agent          "/bin/sh  -
c '/opt/bi…"   2 hours ago       Up 2 hours                             proxy-
agent
8080/tcp    payload-container
```

4.3.4 Run shell commands on the remote container

To switch between two available configurations, choose one of the lines below to copy to the `:PROPERTIES:` drawer for this section.

```
:header-args: :results output verbatim replace :session notebookscontainer :dir /ssh:notebooks-
vm.us-central1-f.quarere|docker:klt-notebooks-vm-cjme:  :exports both  :eval never-
export
:header-args: :results output verbatim replace :session notebookscontainer :dir /ssh:notebooks.u
central1-c.quarere|docker:payload-container:   :exports both   :eval never-export
```

In order to connect to the remote host followed by the docker container we specify the directory as `ssh:notebooks-vm` (including the extra details we got from `gcloud compute ssh-config`) followed by `docker:containername` where we got the container name from running `docker container ls` on the remote machine.

```
echo $JUPYTER_PATH
```

```
$
```

```
head -3 /proc/self/cgroup
```

```
12:cpuset:/docker/4f31864fade2d620150f5bdb8f162bfefd6528670dbe3769b7773570931445ff
11:freezer:/docker/4f31864fade2d620150f5bdb8f162bfefd6528670dbe3769b7773570931445ff
10:cpu,cpuacct:/docker/4f31864fade2d620150f5bdb8f162bfefd6528670dbe3769b7773570931445ff
```

Check the working directory and the list of jupyter kernels

```
(push "-e" docker-tramp-docker-options)
(push "-e" "JUPYTER_PATH=/home/jovyan/.local/share/jupyter:/usr/local/share/jupyter:/usr/share
tramp-docker-options)
```

```
echo $JUPYTER_PATH
```

```
jupyter kernelspec list
```

```
Available kernels:
  ir           /home/jovyan/.local/share/jupyter/kernels/ir
  julia-1.5    /home/jovyan/.local/share/jupyter/kernels/julia-1.5
  maxima       /home/jovyan/.local/share/jupyter/kernels/maxima
  python3      /usr/share/jupyter/kernels/python3
```

If you try to make use of an existing session on the docker container to run one of the `emacs-jupyter` kernels, you find that there is a different usage of the TRAMP remote path specification in the `:dir` property for the `sh` language of babel and with the `:session` property in the `emacs-jupyter` *language* of babel. This is the error I got the first time I tried this with the TRAMP remote path specification in `:dir`:

```
: FileNotFoundErrorTraceback (most recent call last)
: <ipython-input-1-d4b8d99aef95> in <module>
:       1 import os
:       2 __JUPY_saved_dir = os.getcwd()
: ----> 3 os.chdir("/ssh:notebooks-vm.us-central1-f.quarere|docker:klt-notebooks-
vm-cjme:/home/jovyan/")
:       4 try:
:       5     get_ipython().run_cell("""x = 'foo'
:
: FileNotFoundError: [Errno 2] No such file or directory: '/ssh:notebooks-vm.us-
central1-f.quarere|docker:klt-notebooks-vm-cjme:/home/jovyan/'
```

1. Run python session on the remote container The default properties that should apply to this section are

   ```
   :header-args: :results output verbatim replace :session notebookscontainer-python :dir /ssh:notebooks-vm.us-central1-f.quarere|docker:klt-notebooks-vm-cjme:/home/jovyan/  :exports both  :eval never-export
   ```

   In order to connect to the remote host followed by the docker container we specify the directory as `ssh:notebooks-vm` (including the extra details we got from `gcloud compute ssh-config`) followed by a `|` and then `docker:containername` where we got the container name from running `docker container ls` on the remote machine.

   ```
   x = 'foo'
   y = 'bar'
   print(x + ' ' + y)
   ```

   ```
   foo bar
   ```

   ```
   x = 1 + 1
   print(x)
   ```

   ```
   2
   ```

5 Which model construction process works as a whole?

We are working in an applied way to build models, starting with data and using existing tools and methods, but without any strong guarantee that we will find the most effective methods right away. So, with these experiments we are investigating the process of "model construction" generally understood. One example is building computational structures from natural language and technical texts.

5.1 Information extraction from SO Q&A items

We are attempting to extract triples from textual Q&A by using a Neural Machine Translation approach.

5.1.1 BACKBURNER Refinining OpenIE approach DEYAN

1. Idea is we need a basic data model from which we can build things

   - User assistants
   - If we have triples we can do interesting subtasks
   - What's a nice little task to solve in workflows w/ text or w/ code?

5.1.2 GPT/BERT

- Are they relevant for this? Or are they just good for prettifying text to do fun tricks?

1. Ontology papers in Stanford

   Pinpointing what can and can't be done. E.g., phrase structure. Linear probes into the model, correlate weight structure of attention nodes and classical phrase extractors. This uses Stanford's phrase structure parser (which is based on correct phrase structure extraction).

   GPT & BERT have the *beginning* of capturing. Classical embedding results hold (King is to Queen...). Pretrained model gets maybe 85% of the classical task.

   So, if we get a downstream task (relatively shallow linguistic task X), then you have an expectation that with 10K examples to fine-tune, you'll get a decent outcome.

2. Deyan's prior notebooks

   Had borderline reasonable results, extracting triples that were linguistically plausible SVO triples. But these weren't business ready. It would get confused about extracting only a fragment ('elephant', not 'pink elephant'). Still, this gives validation that concept extraction works.

3. Play with rewrite rules?

   - Michael Kohlhase's thesis deals with this (using unification)
   - In language land, rewrites are just translations
   - In ML, it's just a model translating things

   In a serious program language setting, Python 3.7, we'll use the abstract syntax tree of Python, doing formal bits

4. Categorise: Q or A?

   - A/B test explanations?
   - "Grammarly" for SO (but this is 2-3 times harder than improving documentation)

5. Extracting metadata

   (a) Given text as question answer, provide as much metadata as possible

   (b) Instead of triples, care about words that may not even be in there

   (c) Google Photos They've used individual classifiers for any label of interest "church", "cheese-cake". They have many NN classifiers, one for each photo.
   This shines b/c you can annotate people with names. If I search for "me plus my parents" I get exactly what I'd want.
   This would be a bit intense if you have 1 million data for 2m gigabytes of address space.

6. Map: Q's to A's and vice versa

7. Identify duplicate answers If someone answers, people don't ask.

8. Identify relevant answers

   (a) CL: match "I'm buying what someone is selling"

   (b) Iterating or recursively doing this as a tree

      - `(A (B (C D E F)))`

      (This is pretty easy to evaluate.)
      E.g. with Wikipedia internal links: do they reference as related or...
      On SO there's a second aspect: "I'm trying to achieve X but I'm failing in this way." The answer is a rewrite. Not a dependency but it's about mastery.
      "Recommending comprehensive solutions for programming tasks by mining crowd knowledge."
      Link text + surrounding context: does the target page link back? And if it does link back they are of mutual importance.
      Context will tell whether it's a general or specific concept.
      JC: Q/A can also be seen as a link.

9. Route questions based on expertise This is something that people have looked at.

10. Why have a man page if you could turn SO into man pages that interact? In general docs are trash, so you google and use SO for tasks. Pandas docs are almost intentionally obfuscated, the examples are useless.

    Competing with Google-for-StackOverflow isn't a great plan

    But could I improve the documentation itself?

    (a) Autogenerate better documentation for python

    - Python is ubiquitous and there are a lot of SO
    - There could also be demand
    - Ontology could turn into TOC for the guide

11. Validating GPT as usable or not usable in...? There's a terminal that uses GPT. You could describe your CSS and it changed an English description into a webpage template.

12. Given a schema it can generate a query. There are text summarisation quips (e.g., generate abstracts).

13. If you extracted information this way we could use STAN to validate a hypothesis E.g. get estimates about sizes of groups on SO.

14. Pie-in-the-sky You could use nonparametric Bayesian models to 'tame' a neural network and make it interpretable. You can put it into an end-to-end differentiable system, alternate generalisable with model structure.

    Tangled Program Graphs

15. "Hate speech" "How do I solve this sort in Python" If I reply enough with Haskell, you can see I'm galling him... this is so much easier in Haskell. You can go w/ stable differences when these 2 user are interacting.

    This is a high-quality answer but in the context of all the answers and questions, you find it's actually hate speech.

    (a) Friendworld It's about frogs that are friendly. This is a Pepe the frog meme. They'd post melancholy or fun frog...
    With interspersed nazi shit.
    Audioplayers can be completely destroyed by playing a certain record.
    If you're looking for honest learning exachanges they are more mundane.
    E.g., account for poor wording.

    (b) BUT... Humans are good at understanding this but computers aren't.
    People were pointing out the subtle stuff, the problem was that there wasn't enough investment to do anything about it.
    In Germany, Twitter filters holocaust denial; even the stuff they (could) detect they don't remove. In the US, if you report it, they'll deny it. (It's a 'prior restraint' thing... it's complicated if you're responding to someone's complaint.)
    Look at two Nazi related words and see if they form a hashtag. `#jewspiracy` etc.
    Filters are however very difficult.

    - An automated white-knight that did the responding for you
    - But they want you to engage...

    You could do tricks, people started using `#proudboys` for something else.

    (c) Example: how does responding to hate speech influence things?
    Study tracking activity and challenges as to whether people continue posting hate speech.

    (d) "Consider writing this in a more assertive way" I wonder if possibly...
    Guess the degree of someone by reading their email

16. Automatically generating docs from type signature Maybe going for a language with static types could be a way to combine free association in the structured data.

    This is more robust than "write language and get code out."

    "Write code with a bug, get SO Q&A back" (Crokage?)

    Starting with working code. How would you generate failing code. How would you generate failing unit tests? (E.g., "fuzzers" that generate near arbitrary run-ti) Put in integers, get output. Generate wrong unit tests.

5.1.3 Overall commments

These are translation or compression style problems.

Code generation demos are pretty suspicious: GPT3 doesn't make off-by-one errors, it uses completely different function syntax.

Like the motivation behind it. Z was recently criticising auto-generation of query program. The amount of time it takes to debug the query.

"Count all the listings" but rather queried the database's AirBnB table. What if there are multiple tables w/ similar names?

If you put leashes on these things, using solid methods.

5.1.4 We didn't get one simple

5.2 Knowledge graph

Once we have a model of knowledge from Q&A items, e.g., in the form of triples. we will want to be able to do something with this material. One way in which it may be useful is in combination with an existing knowledge graph. For example, we can look at material from Concept Net. We may also have to make some of our own Concept Net-like graphs.

5.2.1 Practical work

We can already take some practical steps here, along the lines of the earlier papers "Modelling the way mathematics is actually done" and "Towards mathematical AI via a model of the content and process of mathematical question and answer dialogues".

5.2.2 **STARTED** Analyse a small sample of examples from s.o. JOE

6 Underlying foundation

We believe that category-theoretic foundations will help us make progress across different representations of code, process, model building, and so on.

6.1 Category theoretic glue

We want to develop enough theory that we can use it to frame our experiments. We are trying to do this in a computationally meaningful way.

6.1.1 Feature: Understand comma categories as a potential "backend" RAY:ZANS

6.2 Probabilistic programming for scientific modelling

Probabilistic programming is useful within both scientific modelling, and, potentially, as part of a program synthesis toolkit.

7 POTENTIAL PRODUCTS

Synthesis of some of our *projects* could lead to marketable *products*.

7.1 Agent model

One of our central intentions is to instantiate our work in an agent model of Q&A and programming. This is based on Alan Turing's suggestion that computers could talk with each other to sharpen their wits.

7.2 Recommender System

We could consume various analyses of Stack Exchange data to make recommendations.

7.2.1 Possible implementation strategy: build on a version of GPT fine-tuned on SO Q&A tasks

Could we set up a simple version of GPT trained on Stack Overflow data, just to get it working? Then think about how to get a learning loop set up to improve the results...

1. Ideas

    - Could this at least help a human navigate the questions on Stack Exchange?
    - Rather than just answering the question, generate the answer and use that to guide search (by combining generation with document similarity)
    - Use a distance to set up a margin of tolerance

2. Precedents

    - Stack Roboflow creates ersatz Q&A using `AWD_LTSM`. Surely we can do better?
    - In Google Books, they use crappy OCR which is good enough for search, but you wouldn't want to read the output. For search, they use something like rewrite distance, finding something 'within 5 errors'.

3. Analogue

    In parsing, it's not just edit distance but has to involve the grammar

4. Case against going too deep:

    - Code generation is hard

5. Case against worrying about that:

    - Worry instead about applications like generating learning packets
        - E.g., learn everything there is to know about `git` from Stack Overflow in a nicely organised way.
        - E.g., compare the Schuam's Outline series: could we reassemble open source clones of Schuam's Outlines by retrieving contents from Math.Stack Exchange?

6. Application of the model: Display SO with similarity graph E.g., use generated answers to help identify 'similarity'.

7. Related work

    - `https://github.com/stared/tag-graph-map-of-stackexchange/wiki` presents a nice-looking map of the relationship between tags.

7.3 Visual Interfaces

7.3.1 Graphical flow for programs

Can we model more general program flow in a similar fashion to Monocl?

7.3.2 Limitations

The idea of graphical programming languages is linked with the Deutsch limit (named for noted programmer L Peter Deutsch, not physicist David Deutsch FRS, though perhaps he could come into play later):

> *The problem with visual programming is that you can't have more than 50 visual primitives on the screen at the same time.*

7.3.3 Automatically create visual interfaces

Here's an idea: assuming we have enough text mining pixie dust (on corpora of linux man pages, and stack overflow questions/forum posts about linux commands), it might be possible to do:

```
user:~$ make-gui-for ls --output ls.py
```

7.3.4 Feature: Build infra for generating and displaying graphs.

E.g., we can generate graphs based on code flow.

```
(defun triangle (n)
  (if (equal n 0) 0
    (+ n (triangle (- n 1)))))
```

This would then be related to the visual code walk through feature described below.

7.3.5 Feature: Visual code walk through

Ray is working on a visual code walk through. This should be seen as another interface to the same basic underlying information, sort of like how Org Roam is the main interface to the data served by Org Roam Server.

1. General evaluation strategy for these demos:

    - *'Would anyone want to use this?'*
    - E.g., in the case of Emacs "learn X in Y" demo.
    - If there is interest, work up to covering the HtDP book

2. Related work

    MAUDE framework. You describe your programming language using rewrite rules in K. They define tools to auto-derive rules in K.

    Program slicing 'Galois connection on the traces'. This allows you to find where bugs appeared. People tend to look in the most recent. Imagine a call-graph of all the variables, so it gives you a minimum trace, showing where your bug can be found.

7.4 Data course

There's a new book available from the group affiliated with STAN. It doesn't go very far, but it has tons of examples. They have data sets about all sorts of stuff. So the idea would be to take, e.g., the notebook on linear regression, and go through...

7.4.1 Idea

Start with a method, then go through lots of examples. Make this consistent with the way we would teach HtDP.

"Here's a data set, here's a method that would make sense to apply."

7.4.2 A quandry

Note that hand-coding of a curriculum vs making a general framework that anyone can contribute to (e.g., to make their own curricula) are pretty different things. We will sort out this ambiguity later.

7.4.3 Sources

There are tons of great data sets, but the issue would be digging into the details of some of them. The real issue is coordinating. We want to start with e.g., intro to linear regression, then hierarchical linear regression, and working up to things like Lotka-Voltera model.

- Datopian

7.4.4 How to build up to this?

- E.g., setting up the pre-requisites of the platform
- Setting up a tutorial on model building in a certain domain, get 10 people in the specialised tutorial, how is it received
- This would start building up the group of people
  - Using someone else's platform would be different from using our own platform
  - Which of these is the focus? (Good question but let's have one or two sprints beforehand to see where things are going.)

7.4.5 Assumptions

- Keep platform open source, assume people would want to use

7.4.6 Comments

- Platform is quite a general word, but in a way we are trying to make something easier
- The platform is just an interface to a piece of technology we build. The core is really on the backend.
- So the focus should be on the backend not on the javascript bits.
- Maybe leverage more existing technologies for the platform, where building it basically means installing it.
- Nextjournal: this looks good because they have UX designers to polish things
- Cloud-based Emacs: Would allow you to back your instantiation as if Emacs is your operating system, 500GB instance on Google Cloud

7.4.7 Status

- Cameron has code to set up a multicluster platform available off the shelf that we can start with
- Ray has been doing similar things for personal use, though if this helps write biology papers.
- What if our user interface was Emacs?
  - Different keybindings; developers like Emacs or Vi...
  - Org Bable exists & we can refer to this for now

7.4.8 Reference

- Michael Betancourt: Towards a principled bayesian workflow

7.5 Paperspace DO NJ etc. Collaboratory

This would be a potential user-facing product in which we could deploy various curricula, share various tools for interacting with scientific/computational models, and build a "knowledge hub" of people who could do scientific work.

8 BUSINESS DEVELOPMENT

8.1 Relationship to purpose

Understanding how the business activities relate to the purpose? We might do things that appear unrelated what we say at Why not what to serve customer needs in the mean time. However, if we do, we should either come up with some reasoning about how this helps us address the purpose, or revise our statement of purpose to reflect the current reality. This presumably isn't hard to do, e.g., we could say "once we have a successful business we will pour $x\%$ into research," but in any case we should clarify this.

8.2 Roughly B2C

- Launch some version of the Emacs Hyper Notebook as a cloud service. (Build it first and test it first.)
- Visual Interfaces: Develop a user interface on top of more advanced data analysis tools. (The focus is on the infrastructure that allows you to convert a graph into a neural network or whatever.)
- Data course (training format): Recruit people to take our course for a fee.
- Paperspace DO NJ etc. Collaboratory (Edtech SaaS): People would build their own courses/projects on our software and pay for licensing.
- Teach arbitrary coding (Edtech SaaS): People would use our tutoring system to improve their programming abilities.

8.3 B2B

- Agent model (software as a service format): We can run our agent model to generate new code or other insights. People can pay for compute plus a premium for quality.
- Probabilistic programming for scientific computing (Consulting format): going around and creating customers by talking to businesses, saying "Using proababilistic programming — or other technologies — we can optimize this, this, this, and this, saving you this much money."
  - Many companies hardly use any AI, let alone deep learning. If you can hustle and sell things, this can work.

  - However, we don't want to sell AI snake oil, so if we are going to do consulting it should be around topics that we're actually experts on. For example, plausibly, we could talk about modelling *documents* and *workflows*.

8.4 Different kinds of users

If we want to build a business, we should focus on who our target users actually are, and what problems we can solve for them. Typically we would build the business in a customer-centric way. So, for example, are the users/customers:

- Advanced STAN users, or,
- People who don't know how to do data analysis but who can make graphs.

Broad categories of users are surveyed in the Downstream.

8.5 Related work

- Be wary of competing with things like Roam, though some level of competition is intrinsic in business.
- "Roam scratches my itches for document and graph aware note taking pretty well."

8.6 Upstream

This is a place to keep track of the upstream tickets we're interested in. Our ability to resolve some of these issues might suggest that we have an ability to deliver things that are useful to other people and would therefore be a step in the direction of BUSINESS DEVELOPMENT.

8.6.1 emacs-jupyter

- :id not set in jupyter-start-kernel #296
- 'Kernel did not respond to kernel-info request' when using ssh session. #191

8.6.2 Firn

- links inside :noexport subtrees shouldn't generate backlinks in their targets #57

9 RESEARCH OUTPUTS

We would like to publish some papers, though as Deyan points out we should only do this when we have high-quality results:

> Deyan: *Every paper that is published for the sake of an academic's publication record, rather than for its scientific merit, is potent fuel for science denialism. The short-term shortcuts for a personal career, when compounded, cause long-term harm to the scientific endeavor.*

So, what can we do without shortcuts?

9.1 Advances in tutoring systems for programming

This would be a survey paper that would inform our efforts to Teach arbitrary coding. Follow references, start with 'AI and tutoring'.

1. (2014) "An adaptation algorithm for an intelligent natural language tutoring system"
2. (2008) "A novel approach for constructing conversational agents using sentence similarity measures"

9.2 Advances in knowledge mining from technical documents

This would be a survey paper that would inform our efforts on *Information extraction from SO Q&A items and the Knowledge graph approach. Note that if we can find survey papers that others have done, that's pretty much just as useful, and saves us a bunch of time.

9.2.1 STARTED Reading "Machine Knowledge" paper DEYAN

9.3 An ABM of the computer programming domain

This would be a paper writing up our agent model work.

The paper could also correspond to a "whitepaper" that talks about how we are able to "mine" computer programs automatically. This would contribute to a long-term business in automated programming (and potentially other kinds of automation work).

10 Bottom

By the time we get to this point, we will have established some impressive research outputs, a potentially profitable business, and a teaching/upskilling platform for technical and scientific topics.

Figure 1: Network view

10.1 Downstream

What do our potential users look like?

10.2 Consulting clients

We discussed the idea of doing consulting for clients who are interested in using scientific models.

- Play through again as a consulting client

10.3 Scientific software developers

We imagine some software developers consuming "tutorial" content we produce, and improving their skills and abilities as a result.

- Play through again as a scientific software developer

10.4 Automated tutoring system users

We imagine some students using AI software we develop. In some cases they could be "students". In other cases, they could already be professional developers.

- Play through again as an automated tutoring system user

10.5 Programmers

We imagine any programmer having some use for our tools. "B2D" (Business to Developer) is an emerging category of enterprise where we can do interesting things.

- Play through again as a programmer

11 Organisational infrastructure

This section is mildly-technical appendix. It looks at our organisational infrastructure itself, including simple things like the technologies we use for communication, and more involved things like "how we communicate" more broadly. (This is a good candidate for splitting off into its own separate wiki, if for no other reason than that it takes up a lot of space in the generated PDF.)

11.1 Schedule and activities

Presently we are meeting 20 minutes a day at 4PM UK time, 11AM Eastern, on Discord for a "coffee chat".

Previously we tried to maintain a schedule of longer meetings (UK evenings):

- Monday: Seminar
- Wednesday: Workshop
- Friday: Studio

That seemed to be too many meetings. Whatever we do about regularly scheduled meetings, we might want to look at how to best pursue of *topics of mutual interest such as:

- Readings on rewriting rules and production systems, and higher-dimensional graph-like things
- Business development around open source, knowledge management, etc.
- Reviewing the value add of Wiki ways of thinking and working, which we have a pretty broad range of experience with
- R&D around 'lenses' in ACT: structure for bi-directional transformations, to enable changes in a projection

So far, this Roadmap has gathered information on some of the topics that have been discussed, but not all of the things that we could see ourselves working on together.

As another activity we may want to get scheduled one or more sessions focused on business stuff.

11.2 Project orientation

Some of this will be different depending on whether we think of this as a "business", or as "a business of some specific nature": primarily centring on "who does this business do business with?"

- Status - where is the project right now?
  - Right now *this overall project* is in a "project development" mode.
  - What are the (multiple) *success indicators* or *proof points* or *failure indicators* for each of the projects? (E.g., going to the casino with $20, you might quit when you get below $10, you might leave when you get above $50.) E.g., need of customers for X, our credibility in X?
  - For the various sub-projects: one relevant thing is "how long is it before thing is likely to make money?" (AKA, "Cross-over.") Or "what else is needed for this to make money?"
  - In particular: maybe take a couple months to see how things are going with a given sub-project? This gives evidence of what we can produce when we work together. We might then ask, who else would care to pay for this?
  - We have listed 4 active projects (`https://miro.com/app/board/o9J_kmPNvaQ=/`); maybe the blog is another one.

- Roles and Responsibilities - *who is handling the standard project roles, and what are they responsible for doing?*
  - Each individual sub-project is likely to have different requirements (e.g., some may need 2 people, some will need 1, etc.)
  - If there's more than one person involved it becomes a parallel architecture
- Goals - *What will this project achieve?*
  - "If I do something valuable, the money will come later."
  - Some of them we might be willing to take the risk of investing time and energy based on whether it looks directly useful to us.
  - Some, like a course, we may need the information about whether it's likely to be taught.
  - Some could become a paper or the building block of a business: these can be small demo projects.
  - Alternatively, in a consulting mode, our role becomes understanding customer goals and helping rationalise work to fulfil them.
- Resource Requirements - *What (people, money, things) are needed to accomplish this project? Where do they come from?*
  - We each individually need some money, but it's not totally clear that the *company* needs some money.
  - If we wanted to replace any one of us with an employee, then we'd have to have some funding source.
  - If the number of person-hours for the goal is quite high, then it's unlikely for the goal to be achieved without funding.
  - E.g., what would we need to be able to do consulting?
- People - *Who are the people working on this project? Who can I ask for more information? How can I best get in touch with them?*
  - If we were to be doing consulting, then it becomes about serving specific customer needs.
- Approach - *What is the overall strategy for accomplishing this project?*
  - Whatever we choose (e.g., consulting vs product development) we should choose it based on some data and analysis.
  - Wherever we are now, the question is what's needed to move ahead.
- Workplan and Timeline - What are the specific tasks needed to accomplish our goals? When might they happen? Who / what / when (in agile, we specify two).
  - Joe needs some job soon!
  - To do consulting we'd need to figure out customer need and credibility
  - To make progress on the AI directions we need some version of all the things up and running!
- Communication Norms - how have the project participants agreed to stay in touch? what, where and how often are regular meetings? Special ceremonies?
  - In 2 months we'll have 2 more months of experience. So we could then assess things.
  - In advance of that, we might start to understand the expections about how we would gather the data.

  - It should be pretty much fun, and if it's not we're kind of doing it wrong?
  - On an ongoing basis we should be able to check whether what we're doing is effectively addressing the goals we have

- Sponsor - *the person who requires the output of the project and has allocated the resources for it (aka Customer in agile)*
  - So far we're all sponsoring our own work on sweat equity
  - While also trying to be helpful & respectful to each other
  - EF was the sponsor at one time
  - Joe provided chips and dip but the event was strictly BYOB... as long as we're here we'll make the best out of. Polka time!
- Project Manager - the person responsible for the drumbeat and tempo of the project, and for its administrative details, including good project management hygiene
- Lead - the person responsible to the Sponsor for making sure the project is accomplished and to the Team for making sure they are able to accomplish the project
  - Ray: project to build bridges between participants (e.g., systems bio, category theory, stats); this is related to the "transdisciplinary design" course
  - Joe: I'm less technically sophisticated
- Team - people working on the project
  - Everyone will have some constraints (like need $40K per year if it takes more than 20 hours per week)

11.2.1 Project Management Hygiene

- set SMART goals (Specific, Measurable, Achievable, Relevant and Time-based)
- understand tasks required to accomplish goals, then set realistic timeline
- create project plan in wiki
- regular, frequent check-ins to iterate plan (goal, priorities, etc.) if necessary
- after-action reviews at the end of project, including reflection/writeup of positives and deltas
- experienced, well-oiled teams requires less strict project management hygiene
- new, less-organized, or heterogenous teams require more attention to careful project management hygiene

11.2.2 TODO Make a list of actual topics of interest — ALL

If we were just doing "content production" we might think of a list of chapters to write, or podcasts to produce. However, maybe those ways of thinking and working don't apply comfortably here.

11.2.3 TODO Make a project analysis of active projects — JOE:RAY

11.3 Technology

Does `https://github.com/orgs/exp2exp/projects/1` conflict, replace, or serve a different function compared with Org mode agenda items?

11.3.1 DONE Figure out Github project(s) vs Org todos JOE:CAMERON

We won't use Github stuff right now. That's better for dealing with a public-facing contribution workflow when we have actual open source things running with contributors.

11.4 Discord server

We set up a Discord server that we're using for our meetings. This invite link should not expire: `https://discord.gg/pArjt4p`

(We also have a Zulip server set up, but currently we're using it less.)

11.5 OBS recordings

We talked about creating asyncronous recordings (screencasts, audio). We also talked about possibly putting the audio recordings into a threaded voice mail forum, but that's a somewhat different application.

11.6 Code sharing platform

For now we have a Github organisation (`https://github.com/exp2exp`), as well as a separate repo that contains these Org Roam notes, among other things. This could potentially be improved or upgraded in various ways.

11.6.1 Comments

- Nextjournal is interesting
- It's like a Jupyter notebook
- It's like Org Bable so you can run code in any language within the same environment
- If I need to add a bash cell to a Julia notebook, it adds a kernel as needed at the run time
- If I install a bunch of libraries, and save the current environment in a docker container, you can import it
- It doesn't yet have an easy way to make an app?

11.6.2 What if you had a browser based version of Org Bable?

- You could have your notebook, backed by the ability to use Emacs

11.6.3 Examples

- Setting up a data science experiment
- Wadler et al. course in Agda in NextJournal
- But you can't easily treat this as 'Org Roam' (no bi-directional things)

11.6.4 Next evolution

We need a basic code sharing platform to get to work. The next evolution might look like what we've been calling the "Emacs Hyper Notebook"? However, some contributors are not interested in using Emacs for everything. And we can't assume that users would be interested in it either!

11.7 Wiki

The public facing version of these notes is available on a simple web interface, created by firn: https://exp2exp.github.io/. This mirrors the contents of our Org Roam directory. Editing is explained below.

We can also view the contents of Org Roam in a linear form as PDF document... or view the currently active tasks using Org Agenda. In the future we may want to have several different "upstream" locations, based on several different small-scale wikis, all feeding into this one location. That's not hard to set up. Contents can also be browsed in a graphical form either with the built in `org-roam-graph` functionality, or by installing Org Roam Server and running `org-roam-server-mode`.

We can potentially improve on all of this further, bulding something like Metacademy. For now, we describe how to use this simple Org Roam based wiki.

11.7.1 Access

Obtain the sources by cloning the repo at https://github.com/exp2exp/exp2exp.github.io.

```
git clone git@github.com:exp2exp/exp2exp.github.io.git
```

(See below for an alternative.)

11.7.2 Mob branch on repo.or.cz

We're mirroring the repo to an environment that allows anonymous commits (without need for further permissioning). If want to contribute anonymously, info on that is here: https://bit.ly/2EQRHEF

You can review commits to the mob branch here: https://repo.or.cz/arxana.git/shortlog/refs/heads/mob

11.7.3 Setup

Install Org Roam if needed (`M-x package-install RET org-roam RET`).

Subsequently, add this to your Emacs configuration:

```
(require 'org-roam)
(setq org-roam-directory (concat "/home/"
                          (getenv "USER")
                          "/exp2exp/"))
(setq org-roam-completion-system 'helm)
(define-key org-roam-mode-map (kbd "C-c n l") #'org-roam)
(define-key org-roam-mode-map (kbd "C-c n f") #'org-roam-find-file)
(define-key org-roam-mode-map (kbd "C-c n b") #'org-roam-switch-to-buffer)
(define-key org-roam-mode-map (kbd "C-c n g") #'org-roam-graph)
(define-key org-mode-map (kbd "C-c n i") #'org-roam-insert)
(org-roam-mode +1)
```

11.7.4 Bonus feature: org-roam-checkout

If you regularly use your own separate Org Roam setup, you can use this simple context switcher to move between the two. Keep track of the various separate Org Roam installations with `org-roam-library` and then switch between them interactively with `org-roam-checkout`.

```
(defvar org-roam-library `(,(concat "/home/" (getenv "USER") "/exp2exp/")
                           ,(concat "/home/" (getenv "USER") "/org-roam/")))

(defun org-roam-checkout ()
  (interactive)
  (let ((ctx org-roam-directory))
    (if (eq (length org-roam-library) 1)
        ;; Still go ahead and set the variable in this case!
        (progn (setq org-roam-directory (car org-roam-library))
             (message "You only have one choice for org-roam-directory defined."))
      (let ((lib (completing-read "Choose a volume: " org-roam-library)))
        (when lib
          (setq org-roam-directory lib))))
    ;; assuming the user changes context, let's also prompt them
    ;; to choose a new file in that context
    (when (not (eq ctx org-roam-directory))
      (org-roam-find-file))))
```

11.7.5 Interaction

Use the `C-c n f` keyboard command to add new disconnected nodes to the graph, or use `C-c n i` to create a page and insert a wiki-style link, like `[[New Page]]`. Follow links with `C-c C-o`. Display the graph structure with `C-c n g`. It may be necessary to run `M-x org-roam-db-build-cache` to get the graph to match reality. Add and commit new or modified files with git, along with `org-roam.db`, and push them to the repo.

11.7.6 Tags

Some of the nodes have `#+roam_tags` set:

code	meaning
HL	High level
CDN	Can do now
LRD	Longer R&D cycle
HD	Has dependencies
RR	Research Review
RO	Research Output
OTS	Off the shelf
SH	Stakeholder
AN	Annex

Some of the files also have a `#+CATEGORY` set.

11.7.7 Pairing

For syncronized browsing and editing with lockstep.el:

```
ssh pair@178.79.174.58
PW: <ASK JOE FOR THE PASSWORD>
emacsclient -a '' -t
M-x lockstep
```

To open up a real-time collaboration (with multiple cursors), use crdt.el, first to serve the buffer:

```
M-x crdt-serve-buffer
```

And then, from your client, to connect:

```
M-x crdt-connect
```

11.7.8 Linearizing

To turn this map into something we can reliably use, let's try to linearize it.

To downsample from Org Roam (save as `~/bin/roam2org.sh` and make it executable):

```
#! /bin/bash

emacs --batch -l ~/bin/downsample-org-roam.el --eval "(combine-files)" "$@"
```

Here are the working parts (save as `~/bin/downsample-org-roam.el`):

```
(defun downsample ()
  "Process an Org Roam buffer for inclusion in a standard Org file.
Changes title to header, and increase indentation of existing headers.
Changes file links to internal links."
  (if (looking-at "^#\\+TITLE:")
      (replace-match "*"))
  (forward-line 1)
  (if (looking-at "^#\\+roam_tags:\\(.*\\)")
      (replace-match ":PROPERTIES:
  :tag:\\1
  :END:"))
  (while (re-search-forward "^\\*" nil t)
    (replace-match "**"))
  (goto-char (point-min))
 (while (re-search-forward "\\[\\[file:\\([^]]*\\)\\]\\[\\([^]]*\\)\\]\\]" nil t)
    (replace-match "[[*\\2][\\2]]"))
  (buffer-substring-no-properties (point-min) (point-max)))

(defun combine-org-roam-files (&rest args)
"Combine a list of files, specified as ARGs.
The files are to be found in `org-roam-directory'."
  (apply #'concat
         (mapcar (lambda (file)
                   (save-window-excursion
                     (find-file (concat org-roam-directory file))
                     (let ((contents (buffer-substring-no-properties (point-min)
                                                                      (point-max))))
                       (with-temp-buffer (insert contents)
                                         (goto-char (point-min))
                                         (downsample)))))
                 (or (car args) (nthcdr 5 command-line-args)))))
```

11.7.9 Backlog

Part of the idea with a backlog is to go from most-doable, starting with work in progress, to least-doable and potentially vague. Here, then, is one approximate linearization that may or may not meet that description!

Note, this is duplicated in the index file, probably for sanity we should pick one and automate the derived version from there!

```
(defvar files-to-combine
'("20200810131435-hyperreal_enterprises.org"
"20200810132653-top.org"
"20200905124558-why_not_what.org"
 "20200909195629-teach_arbitrary_coding.org"
 "20200810135851-how_to_design_programs_with_if.org"
"20200905124405-construct_critique_improve_models_of_the_creative_process.org"
  "20200905125342-emacs_hyper_notebook.org"
  "emacs_jupyter_remote_debugging.org"
"20200905125023-which_model_construction_process_works_as_a_whole.org"
 "20200905131027-information_extraction_from_so_q_a_items.org"
"20200905131918-knowledge_graph.org"
"20200905124432-underlying_foundation.org"
 "20200905125713-category_theoretic_glue.org"
 "20200905131656-probabilistic_programming_for_scientific_modelling.org"
"20201003205523-potential_products.org"
 "20200905130423-agent_model.org"
 "20200817172825-recommender_system.org"
 "20200810135457-visual_interfaces.org"
 "20200814203551-data_course.org"
 "20200905132603-paperspace_do_nj_etc_collaboratory.org"
"20200814210243-business_development.org"
"20200905134325-research_outputs.org"
 "20200810135325-advances_in_tutoring_systems_for_programming.org"
 "20200810135403-advances_in_knowledge_mining_from_technical_documents.org"
 "20200905132334-an_abm_of_the_computer_programming_domain.org"
"20200906003704-bottom.org"
 "20201003164408-downstream.org"
 "20201003165500-consulting_clients.org"
 "20201003170312-open_source_developers.org"
 "20201003170333-tutoring_students.org"
 "20201003171011-programmers.org"
"20200810135126-organisational_infrastructure.org"
 "20200810135619-discord_server.org"
 "20200811185435-obs_recordings.org"
 "20200814193042-code_sharing_platform.org"
 "20200912223428-wiki.org"
 "20201003164100-forum.org"
 "20200814195259-blog.org"
"sfi/sfi.org"
 "sfi/gather_data_via_stack_exchange_apis.org"
 "sfi/argumentation_theoretic_analysis.org"
 "sfi/process_model_analysis.org"
 "sfi/ml_nlp_bootcamp.org"
 "sfi/initial_ml_baseline_e_g_match_q_a.org"
 "sfi/hierarchical_ml_for_content_extraction.org"
 "sfi/active_inference_bootcamp.org"
 "sfi/agent_modelling_and_sandbox_setup.org"
 "sfi/curate_koans_and_develop_solver.org"
 "sfi/study_with_crowdsourced_exercises.org"
 "sfi/study_with_agent_written_questions.org"
 "sfi/publication_ijcai.org"
)
"An ordered list of files to combine in our export.
```

```
This is where the order of presentation in the downstream org file
and derived PDF is defined.")
```

To combine the files, run:

```
(combine-org-roam-files files-to-combine)
```

To get the indicative nesting (shown by spaces above) to be replicated at the org level, run the following at the top of the exported compilation:

```
(defun indent-org-roam-export ()
  "Utility function to increase indention for selected trees."
  (org-map-entries (lambda ()
                     ;; don't demote the top level items and their sub-items
                     (let ((tag (org-entry-get nil "tag")))
                       (if (and tag (string= (car (split-string tag)) "HL"))
                           (progn (org-end-of-subtree)
                                  (setq org-map-continue-from (point)))
                         (org-do-demote))))
                   nil 'file))
```

Lastly, to rebuild the PDF, all of this can be done with one swift action.

```
(defun rebuild-org-roam-pdf ()
  "Build an org file and PDF compiling `files-to-combine'."
  (interactive)
  (save-excursion (find-file (concat org-roam-directory
                                     "/manual/combined.org"))
    (goto-char (point-min))
    (search-forward "# IMPORT")
    (let ((beg (point)))
      (delete-region (point) (point-max))
      (insert "\n" (combine-org-roam-files files-to-combine))
      (goto-char beg)
      (indent-org-roam-export)
      (org-latex-export-to-pdf))))
```

11.7.10 Publishing to the web

Publishing with Firn is simple:

```
firn build
```

Then commit and push.

11.7.11 Reviewing progress

Something like the following should be all that's get a high-level overview of progress on active tasks, sourcing information directly from the Org Roam files. Add the following to your emacs initialisation script (e.g., `~/.emacs`), evaluate it, and then run `C-c r` to load up the fun. This may not be the perfect presentation yet but it gives an idea.

```
(setq org-todo-keywords
      '((sequence "TODO" "STARTED" "BLOCKED" "BACKBURNER" "FROZEN"
                  "|" "DONE" "DEFERRED" "WONTFIX")))

(setq org-agenda-sorting-strategy '((todo todo-state-down category-down)))

(setq org-agenda-files '("~/exp2exp/"))

(defun org-scrum-board ()
  (interactive)
 (org-todo-list "TODO|STARTED|BLOCKED|BACKBURNER|FROZEN|DONE|DEFERRED|WONTFIX"))

(global-set-key (kbd "C-c r") 'org-scrum-board)
```

This view can then be further filtered by regexp (e.g., your name) by pressing `=`.

11.7.12 DONE Package downsamping code separately JOE

11.7.13 WONTFIX Update the repo instructions to reference this file JOE

11.8 Forum

We talked about using Wikum as a forum, because we liked the idea of a workflow based on summarising discussions. There's now a demo instance set up that we can use, here:
`http://wikum.org/visualization_flags?id=590&owner=holtzermann17`

11.8.1 Could we incorporate the ideas directly in Org or Org Roam?

Perhaps we could incorporate some Wikum ideas right into the wiki here. The idea would be to treat the top paragraph on each page as a summary, and then add discussion threads below. We'd want some system of tags that indicated whether the summary was validated or now. (Note the the original WikiWikiWeb did not have separate talk pages! I don't know if they practiced robust summarisation, either.)

1. REMARK JOE This is an "inline task," via `(require 'org-inlinetask)`. There doesn't seem to be support for nested or threaded tasks, but maybe we would have use for non-threaded forum discussions at the end of any page in the Wiki. Incidentally, for those curious, the formatting of the LaTeX export is controlled by `org-latex-format-inlinetask-function`.
2. END

11.8.2 TODO What might our summarisation workflow look like? ALL

Since we're pretty actively updating our Discord and pretty happy using it, maybe people who are working on Active Projects would be willing to summarise on the wiki, say, weekly? And contribute to a monthly group blog post?

11.9 Blog

This is a public window on our experiments, available at https://exp2exp.com.

Presently, we're still figuring out what the work flow and contents of the blog will look like. The kinds of people to whom we wish to appear credible are described in Downstream, and presumably whatever we put online should match what we think they will want to know.

Zans: *If I implemented as I read things, it would be a pretty interesting blog. There could be a huge market of people interested in following this, this would give a pool of people who know who we are. This is a nice goal b/c it doesn't focus on the product... but it's a deliverable, made up of smaller deliverables, and a concrete benefit.*