Types of Intellectual Property

- **Patent**
  - Rights arise from filing and prosecution
  - Fixed term monopoly on the invention

- **Trade Secrets**
  - Rights arise from operational measures
  - Protects from misappropriation as long as operational measures are maintained

- **Copyright**
  - Rights arise from authorship
  - Protects from copying for fixed (but regularly extended) term

- **Designs**
  - Rights arise from filing and prosecution
  - Exclusivity with respect to look and feel for a fixed term

- **Trademark**
  - Rights arise from use and/or filing
  - Exclusivity of brand for a category of product during continuous use
Multiple Types of IP for a Single Project

- Patent directed to overall system architecture (e.g., data processing pipeline)
- Patent directed to algorithm (e.g., feature extraction)
- Input or output data maintained as Trade Secret
- Copyright in source code implementing the system
- Design directed to unique display format
- Trademark protection of product name
Utility Patent

- Functional aspects of invention
- Term: 20 years from Effective Filing Date

Design Patent

- Ornamental aspects (look and feel)
- Term: 15 years from issuance
When is an Idea Patentable?

- Although you need not have built the invention yet, you need to have enough information to do so.
  - Identifying an interesting avenue of research isn’t enough.
  - Identifying a desirable result isn’t enough.

- New, Useful, Non-Obvious
  - Useful is Easy
  - New is Easy (but beware of your own prior publications)

- Non-Obvious (Inventive Step) is Hard
  - Would someone of ordinary skill have come up with the idea?
  - How much experimentation would it take to make the idea work?
  - “Obvious” ideas include: predictable optimizations, substitution of known techniques
Important Categories of Software Inventions

- **Algorithms**
  - *e.g.*, feature extraction, named entity recognition, clustering, retrosynthesis

- **System Architecture or Overall Processing Pipelines**
  - *e.g.*, data mining techniques, multistage learning, ensemble models, trainable feature extractors, computer-aided diagnosis

- **Neural network configurations**
  - *e.g.*, LSTMs, CNN and RNN architectures
How to Frame a Software Invention

- You don’t patent a result
  - You patent a process, a machine, or a composition giving rise to a result
  - The benefits are important, but the invention relates to the features that give rise to the benefits

- Is this a technical solution to a technical problem?

- Drill down below “but with”
Avoid Abstraction

- "We use a neural network to predict phenotype from sequencing data."

- "To predict phenotype, our system uses a pair of neural networks. The first is a convolutional neural network with a fully convolutional layer, which is trained to predict relevance of SNPs to a given phenotype of interest. The second is feed-forward network trained on the most relevant SNPs to predict phenotype."

- "Instead of sorting the data by hand, we use a classifier."

- "We have designed an ensemble model using cascading SVMs for hierarchical classification."
The Software Patent Pendulum

- *Alice* (2014) and Financial Inventions

  - “integrated into a practical application”

- Improvements in efficiency, runtime, and storage or memory bandwidth provide strong arguments, particularly when dealing with large datasets
Working with a Team

- Tap into the development process
  - Schedule in-team assessment to coincide with project milestones
  - *e.g.*, at the end of each sprint

- Memorialize decision trees and forks in the road of the technology development
  - Consider making this a living document

- Diligent record-keeping helps with accurate determinations of inventorship
  - Inventorship and authorship are different analyses
The use of open source toolkits does not make an invention unpatentable
- If the innovation is a unique processing pipeline, some or all of the components may be off-the-shelf
- If the innovation is a neural network architecture, the fact that it is implemented using a standard ANN toolkit doesn’t render it unpatentable
- But, straight-forward applications of off-the-shelf components may be “obvious”

However, an open source license imposes restrictions on use (which in some cases may impact patent rights)
- Linking to a library, or using a tool is generally low risk
- Modifying open source code, or integrating open source code directly into your codebase is high risk
- However, the specific restrictions and requirements are determined by the individual license, so seek advice if in doubt
Contributions to Open Source

- An open source license is still a license
  - The choice of license dictates the restrictions imposed on users.
  - *e.g.*, requiring derivative works be licensed on open source terms

- Licensing on open source terms does not preclude licensing on other terms
  - *e.g.*, separate commercial and non-commercial license terms

- Licensing on open source terms does not preclude patenting
  - but it may significantly limit the value of the patent depending on the license terms
Questions?

Erik Huestis
ehuestis@foleyhoag.com