Virtual Reality Hand Pose Classification with TensorFlow Estimators

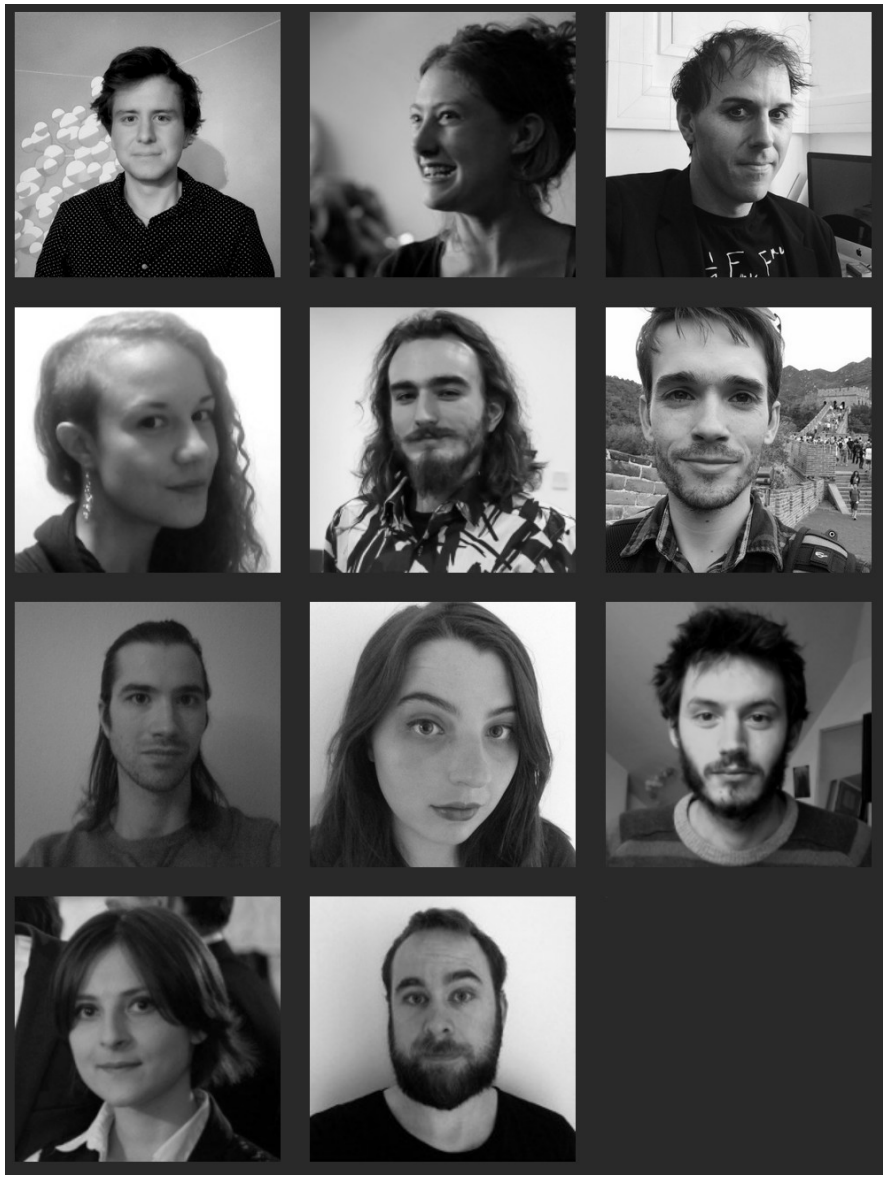
Mike O’Connor
Dept. of Computer Science & School of Chemistry
mikeoconn.org

This talk

- A little bit about molecular simulation in VR
- VR Gloves, and the need for gesture detection.
- Building a classifier with TensorFlow Estimators.
- An end-to-end prototype for a machine learning application.
- Disclaimer: I am not a TensorFlow expert by any means!

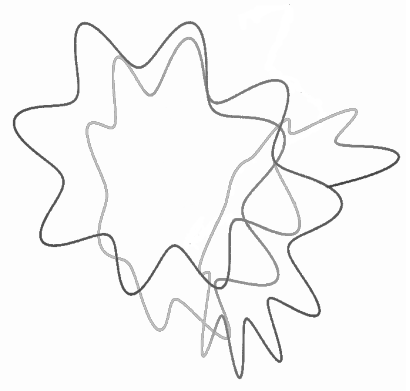
Code: github.com/mikeoconnor0308/tensorglove

iSci



We span a lot of disciplines...

Super-
Computing

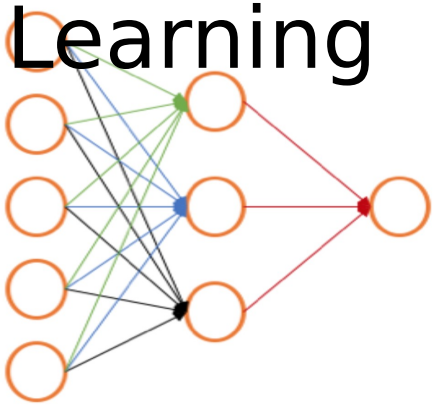


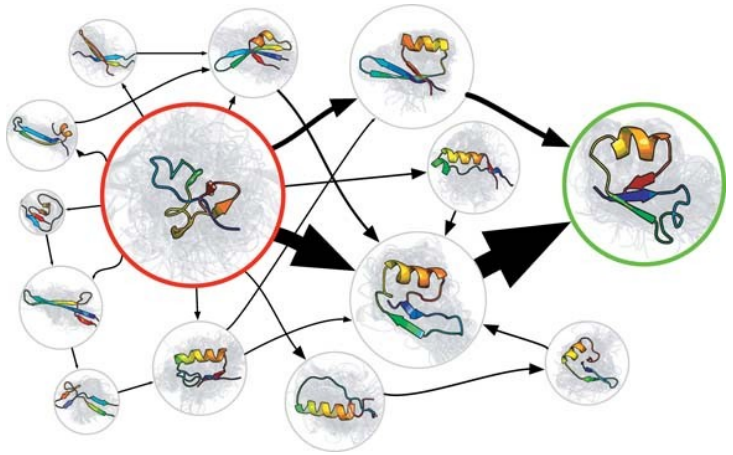
Digital
Arts

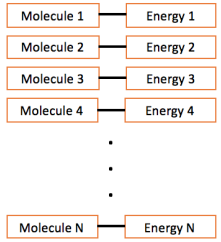
Molecular
Dynamics

Human
Computer
Interaction

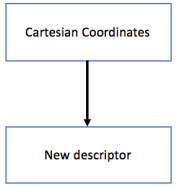
Machine
Learning



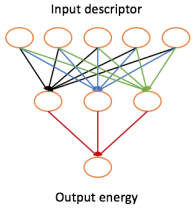




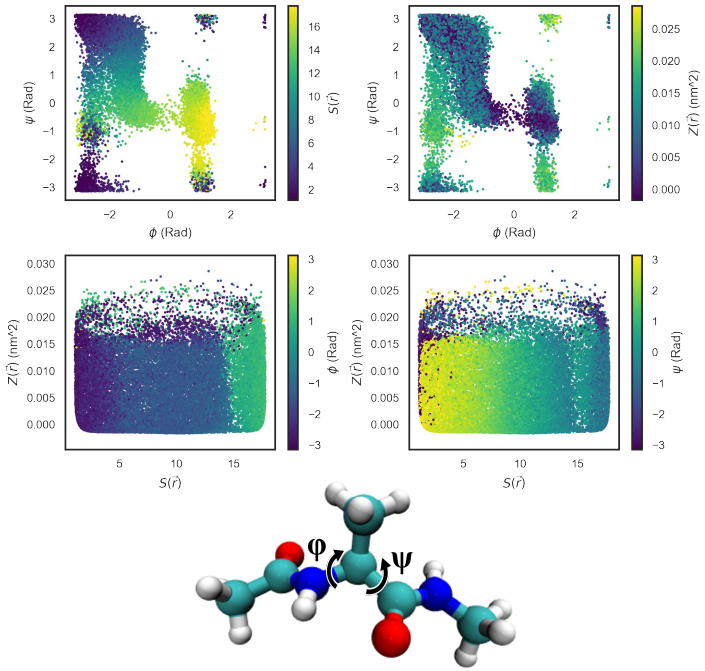
Training data



Convert data to suitable input

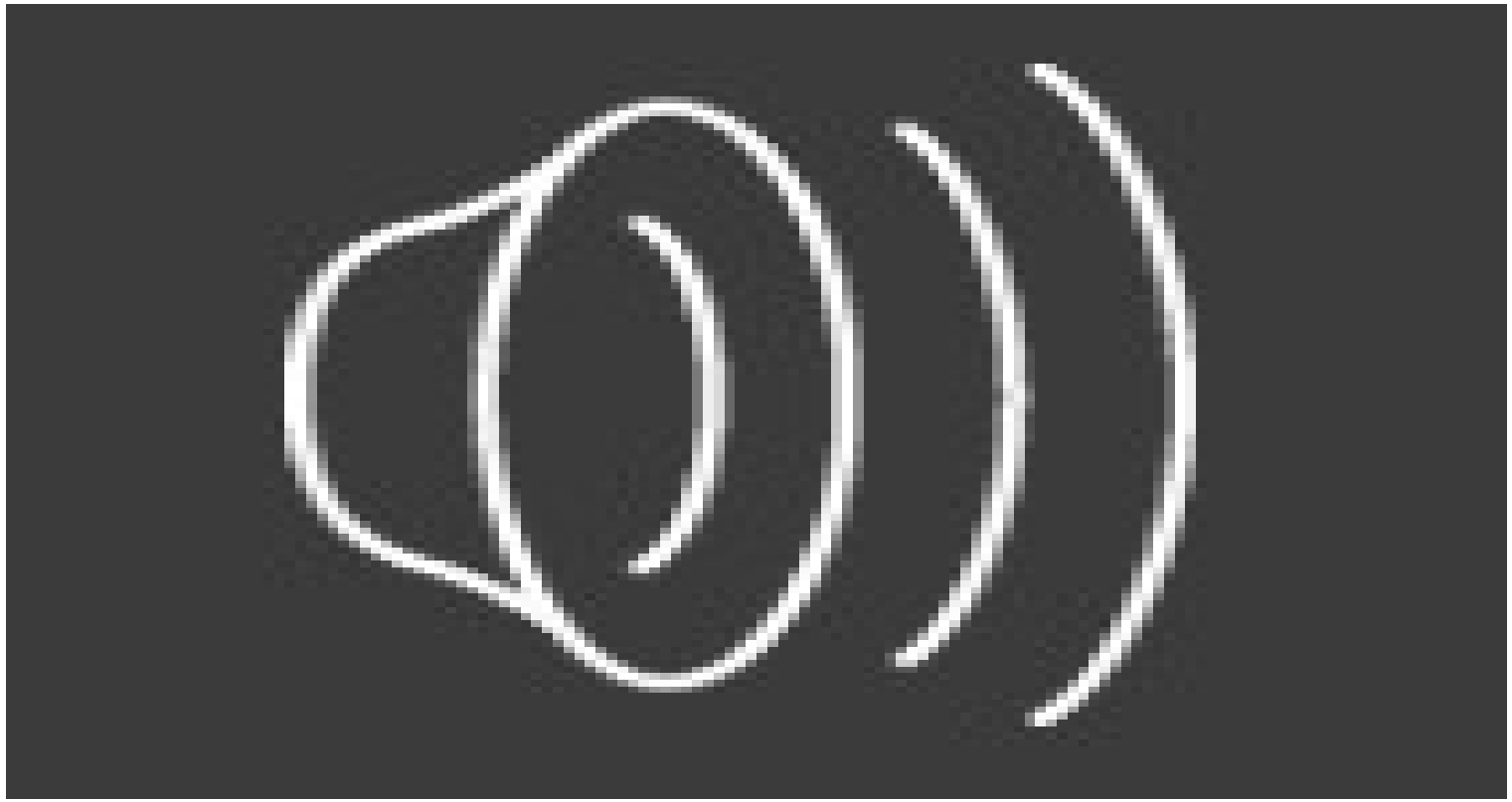


Train neural network



Open source data tools make science happen.

Interactive Molecular Dynamics in VR



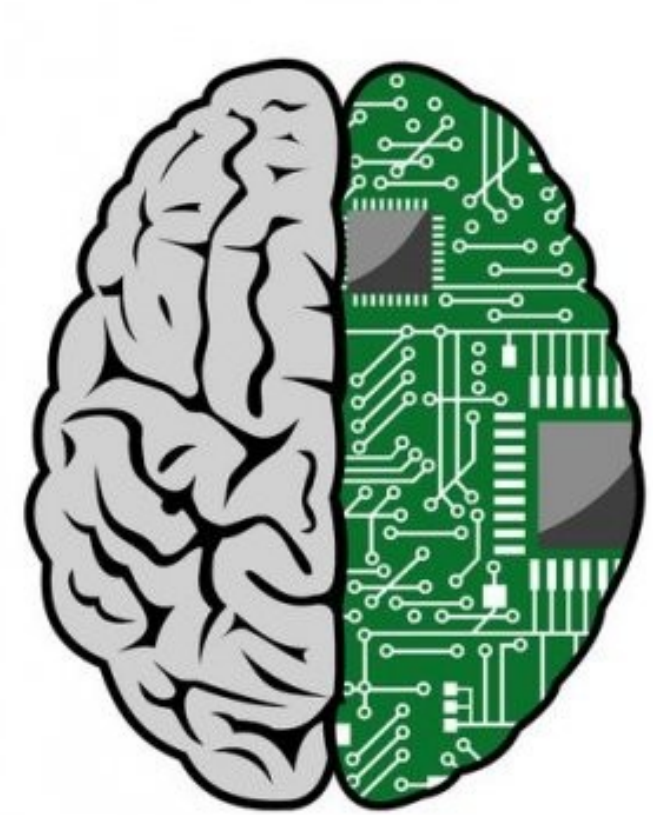
O’Connor, M. et al, *Science Advances*, *4*(6), 2018

Give it a go!

https://isci.itch.io/nsb-imd

Scientific ‘gamification’

Casting scientific problems as games that scientific players can solve



Which computer is faster?

Can we design algorithms that let machines learn from the experts?

Wouldn’t it be great to use our hands in VR?

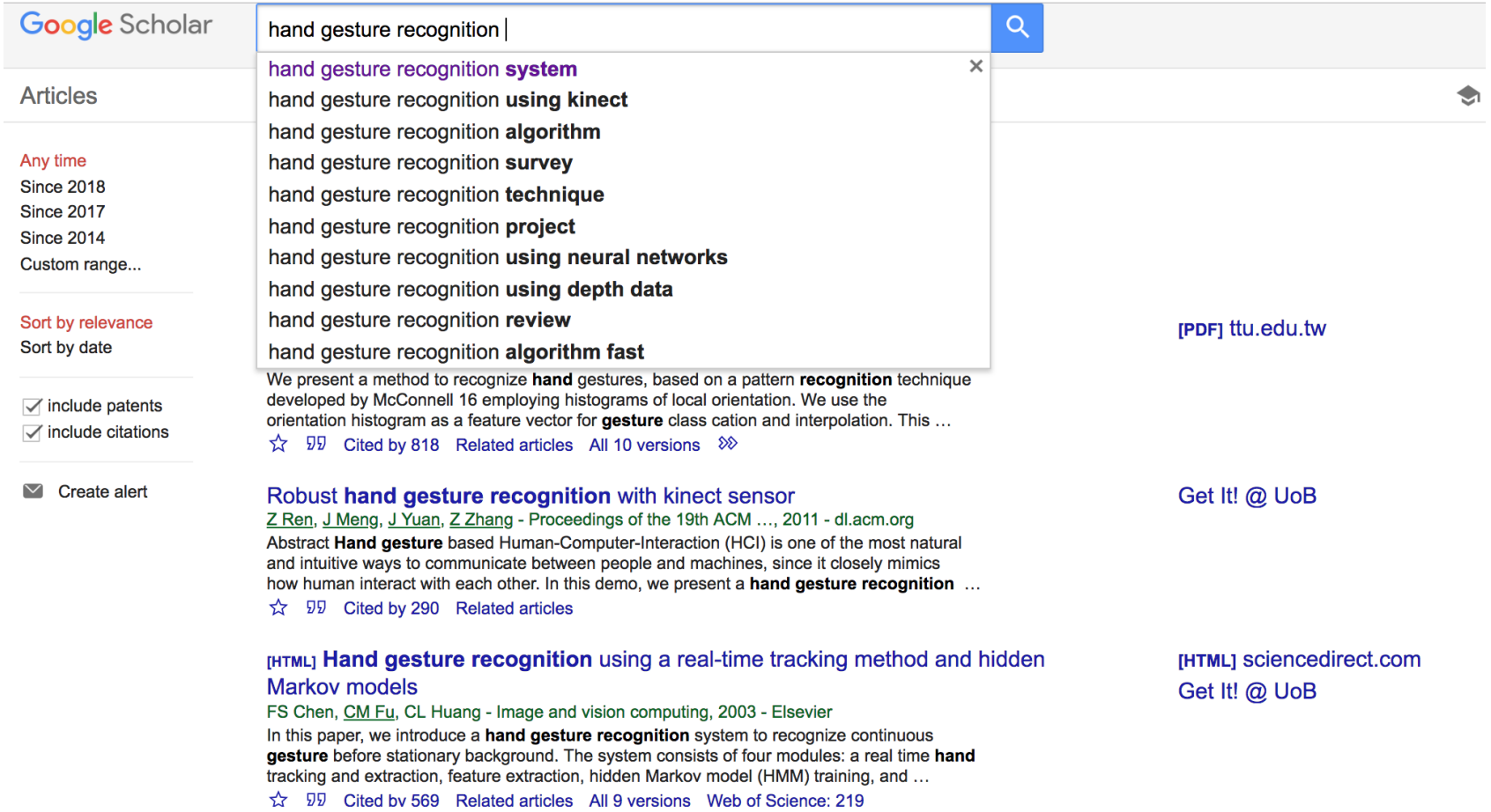
VR gloves are now commercially available

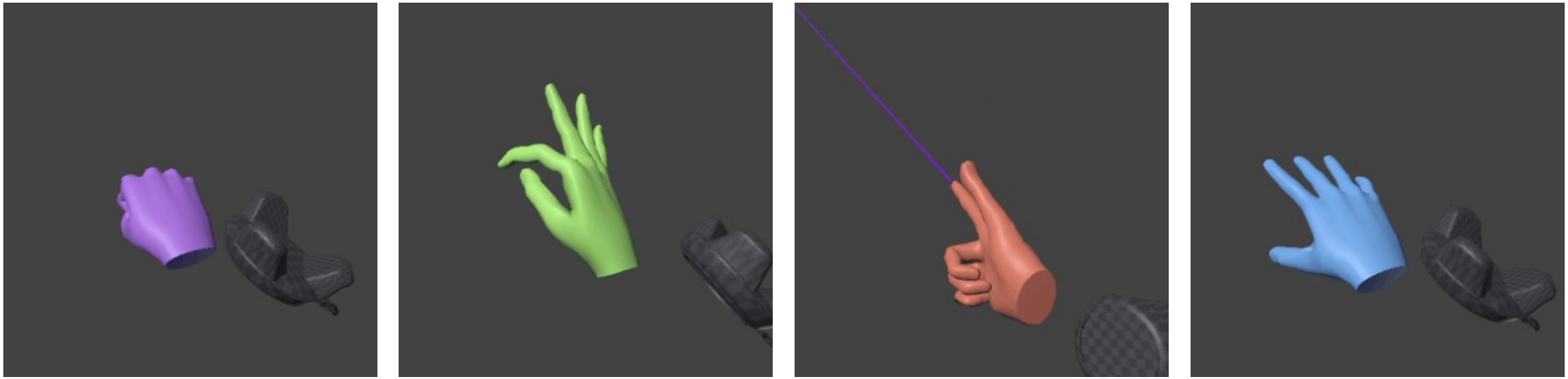
The Glove Demo Scene

Tensorflow* Hackathon

- No gesture controls in the SDK!
- Let’s build a classifier that detects hand poses based on the glove data.
- 2 days – aim for end-to-end prototype.
- Use the Premade Estimators to speed things up.

*Other Methods Are Available.

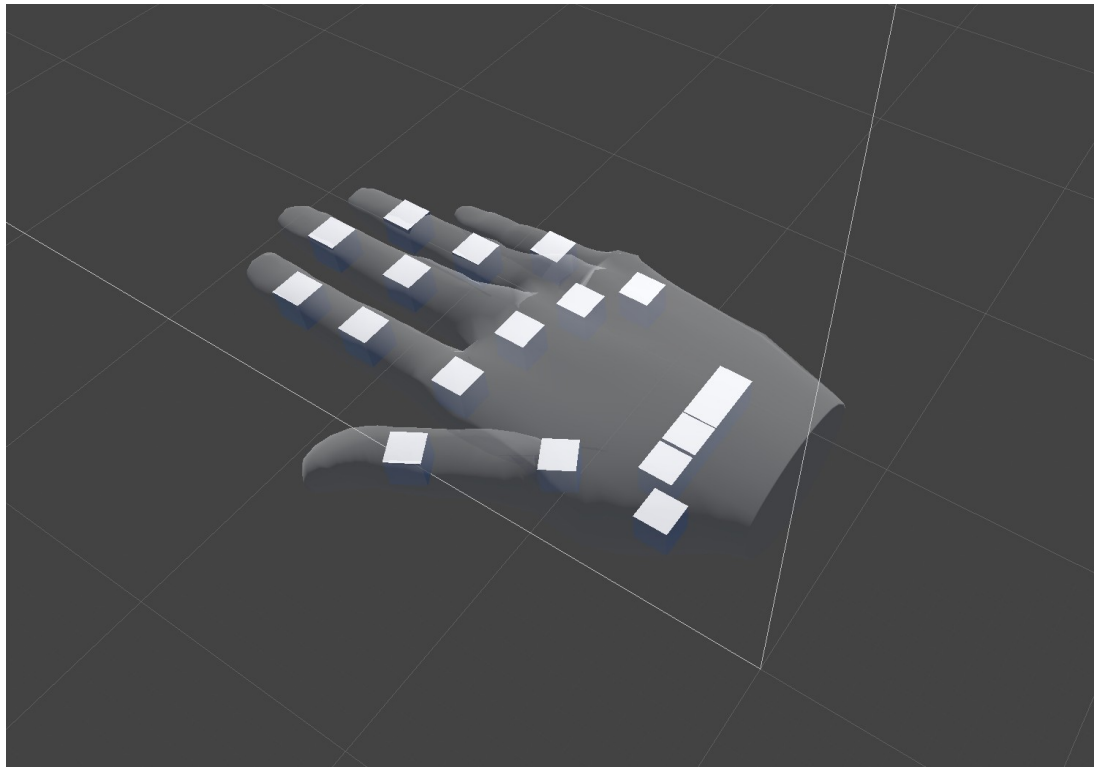




Data Collection

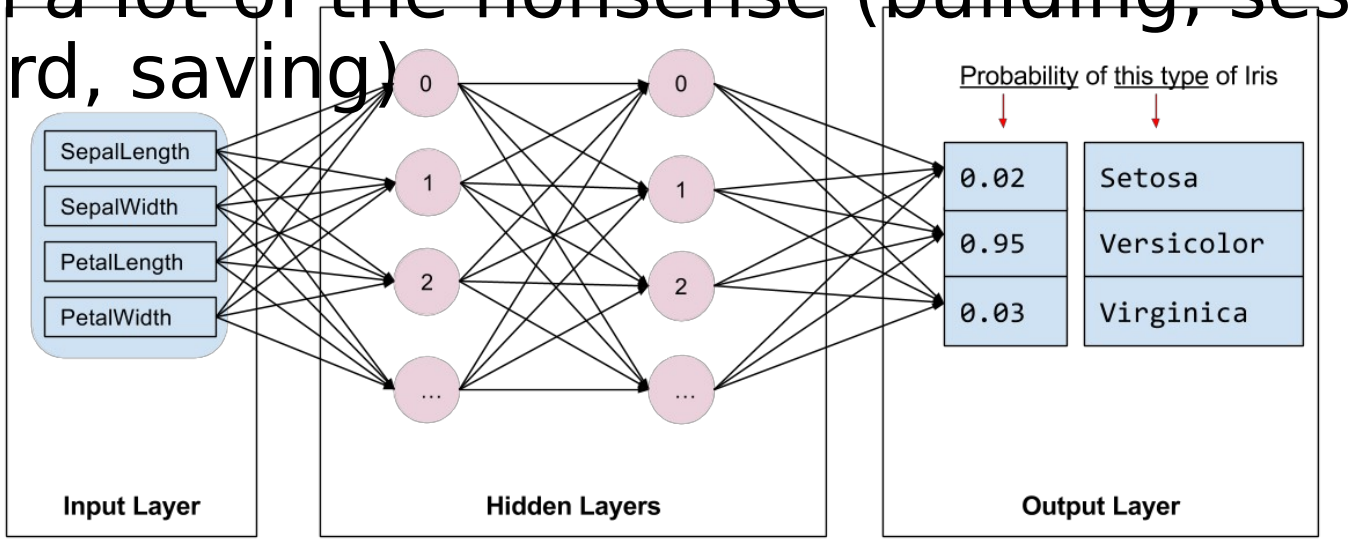
Feature Selection (the dumb way)

- Gloves only provide rotation data for each joint.
- 21 Quaternions => 84 floats.
- No access to raw sensor data.
- We’re building a classifier on top of a model!!



Tensorflow Estimators

- High-level representation of a model.
- A midway point before high level API such as Keras.
- Pre-made estimators for common tasks: classification, regression.
- Deals with a lot of the nonsense (building, session, tensorboard, saving)



https://www.tensorflow.org/guide/premade_estimat

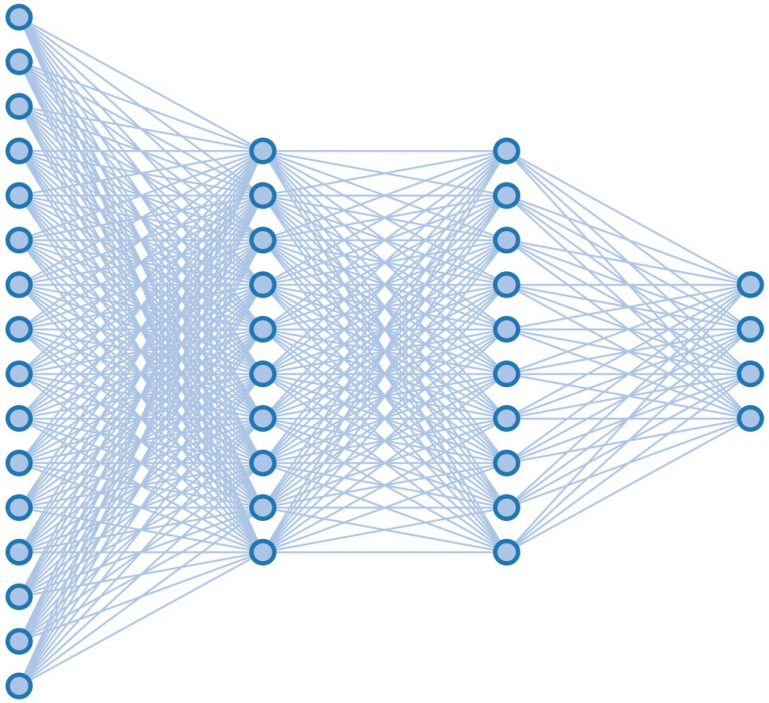
Out-of-the-box model.

```
# Feature columns describe how to use the input.
my_feature_columns = []
for key in train_x.keys():
    my_feature_columns.append(tf.feature_column.numeric_column(key=key))

# Build 2 hidden layer DNN with 10, 10 units respectively.
classifier = tf.estimator.DNNClassifier(
    feature_columns=my_feature_columns,
    # Two hidden layers of 10 nodes each.
    hidden_units=[10, 10],
    # The model must choose between 4 classes.
    n_classes=4,
    model_dir="model")

# Train the Model.
classifier.train(
    input_fn=lambda: glovedata.train_input_fn(train_x, train_y,
                                              args.batch_size),
    steps=args.train_steps)

# Evaluate the model.
eval_result = classifier.evaluate(
    input_fn=lambda: glovedata.eval_input_fn(test_x, test_y,
                                             args.batch_size))
```



Success!!

- 96% Accuracy on test set. (80/20 split).
- Now, we just need to get the predictions to the VR!

Serving a Tensorflow Model: A Survey

Option 1) Use the Estimator.predict method.

```
predictions = classifier.predict(
    input_fn=lambda: glovedata.eval_input_fn(predict_x,
                                             labels=None,
                                             batch_size=args.batch_size))
```

- Convenient, but…
- By default, reloads model every call.
- How to communicate this to the VR app?

Option 2) Use TensorFlow serving.

Serving a TensorFlow Model

This tutorial shows you how to use TensorFlow Serving components to export a trained TensorFlow model and use the standard tensorflow_model_server to serve it. If you are already familiar with TensorFlow Serving, and you want to know more about how the server internals work, see the TensorFlow Serving advanced tutorial.

This tutorial uses the simple Softmax Regression model introduced in the TensorFlow tutorial for handwritten image (MNIST data) classification. If you do not know what TensorFlow or MNIST is, see the MNIST For ML Beginners tutorial.

The code for this tutorial consists of two parts:

- A Python file, mnist_saved_model.py, that trains and exports the model.
- A ModelServer binary which can be either installed using Apt, or compiled from a C++ file (main.cc). The TensorFlow Serving ModelServer discovers new exported models and runs a gRPC service for serving them.

Option 2) Use TensorFlow serving.

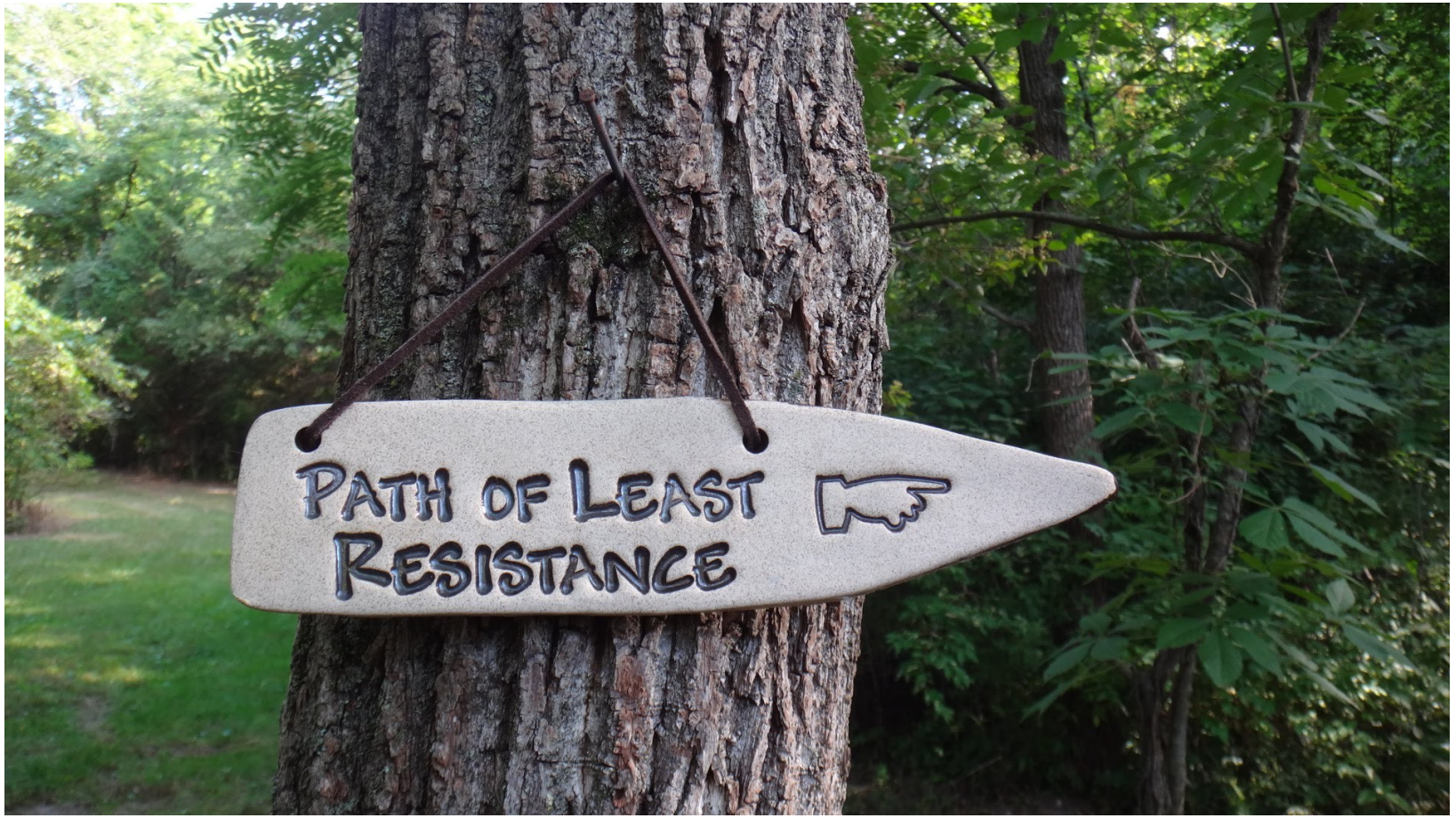
- This is the way to do it for production.
- Images not available for Windows. I’d have to compile it all.
- Or run a container. Again, not great on Windows (no GPU).
- Requires gRPC communication from the client.
- Seems like overkill for this hack!

Option 3) “Freeze” the model

Option 3) “Freeze” the model.

- Freezes a graph by converting all Variables to fixed values.
- Can then be used by C/C++/C# APIs.
- This is how the Unity3D/TensorflowSharp plugins work.
- A good option, but figuring out how to set the inputs/outputs can be fiddly.

I chose Option 1



Communication: Open Sound Control (OSC)

/predict, X, Y, Z, W ....

```
from pythonosc import osc_server

dispatcher.map("/predict", self.predict)
server = osc_server.ThreadingOSCUDPServer(
    (“localhost”,54321), dispatcher)
```

It works!! But it’s really slow...

THREE WEEKS LATER...

Physical and Theoretical Chemistry Laboratory

4.5 ★★★★½ · 2 reviews

Laboratory

Directions

SAVE NEARBY SEND TO YOUR PHONE SHARE

South Parks Road, Oxford OX1 3QZ

QP5W+JP Oxford

physchem.ox.ac.uk

01865 275400

Opens at 08:00

Claim this business

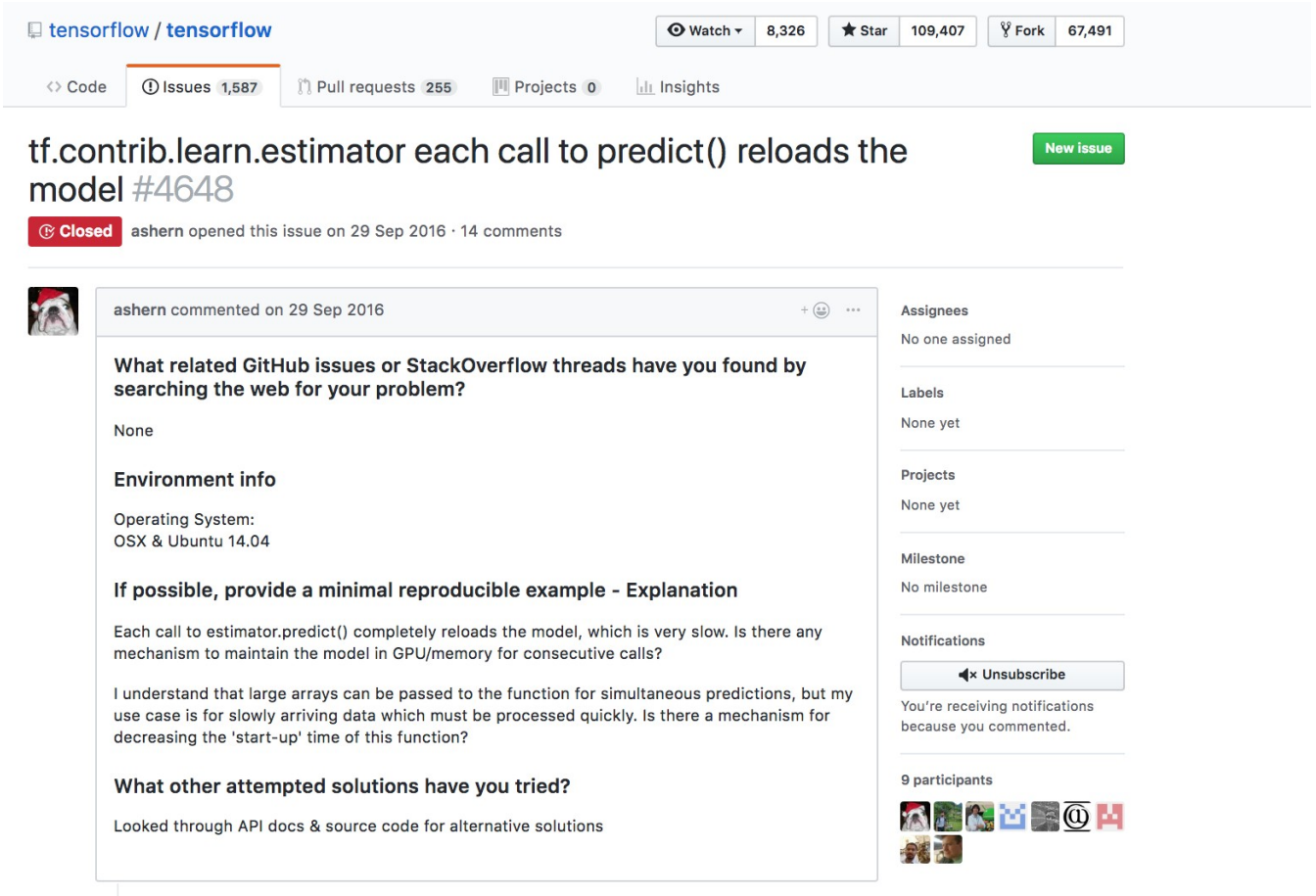
SUGGEST AN EDIT

TMCS

1 more day: Some Hyperparameter Optimisation

- More training data.
- Use 5-fold cross validation.
- Perform a grid search on hidden layer sizes and learning rate.
- Hidden layer sizes now 18 and 20.
- Increased to 99% accuracy. Overfitted!
- More sophisticated optimisations out of scope.

Faster Prediction: Saved by GitHub issue board



Faster Prediction: Saved by GitHub issue board

marcsto commented on 6 Jul 2017

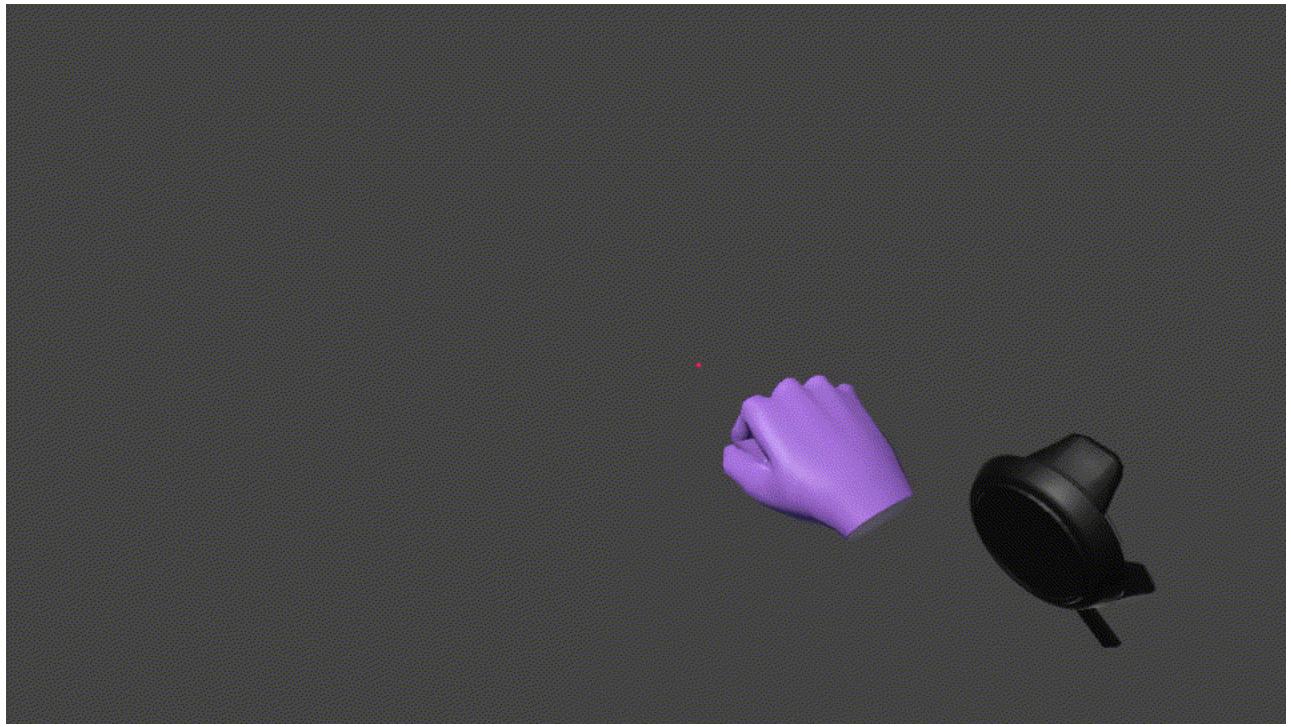
In case someone else runs into this, predict can take a generator as a parameter.

Here's a class that wraps an estimator with a generator. This means you can make multiple calls to predict without reloading the graph.

https://github.com/marcsto/rl/blob/master/src/fast_predict.py

👍 1

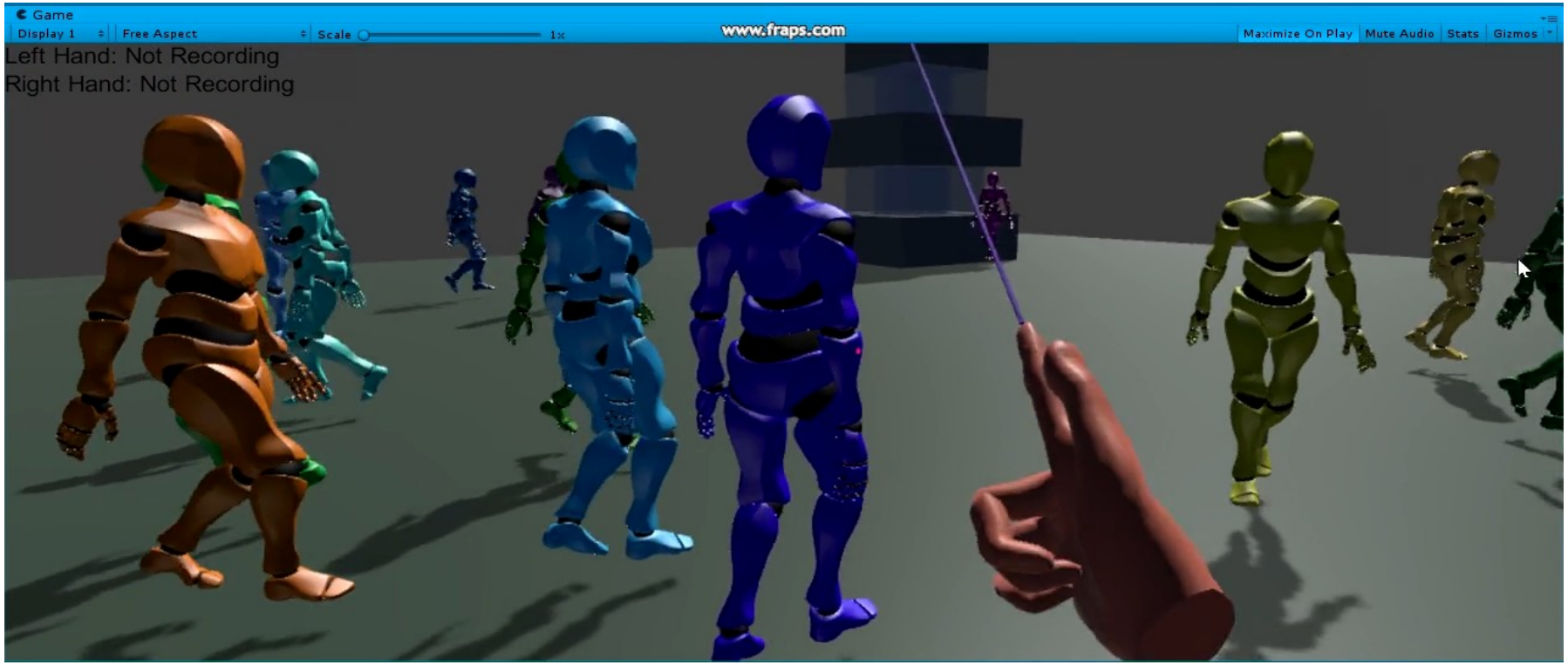
Putting it all together



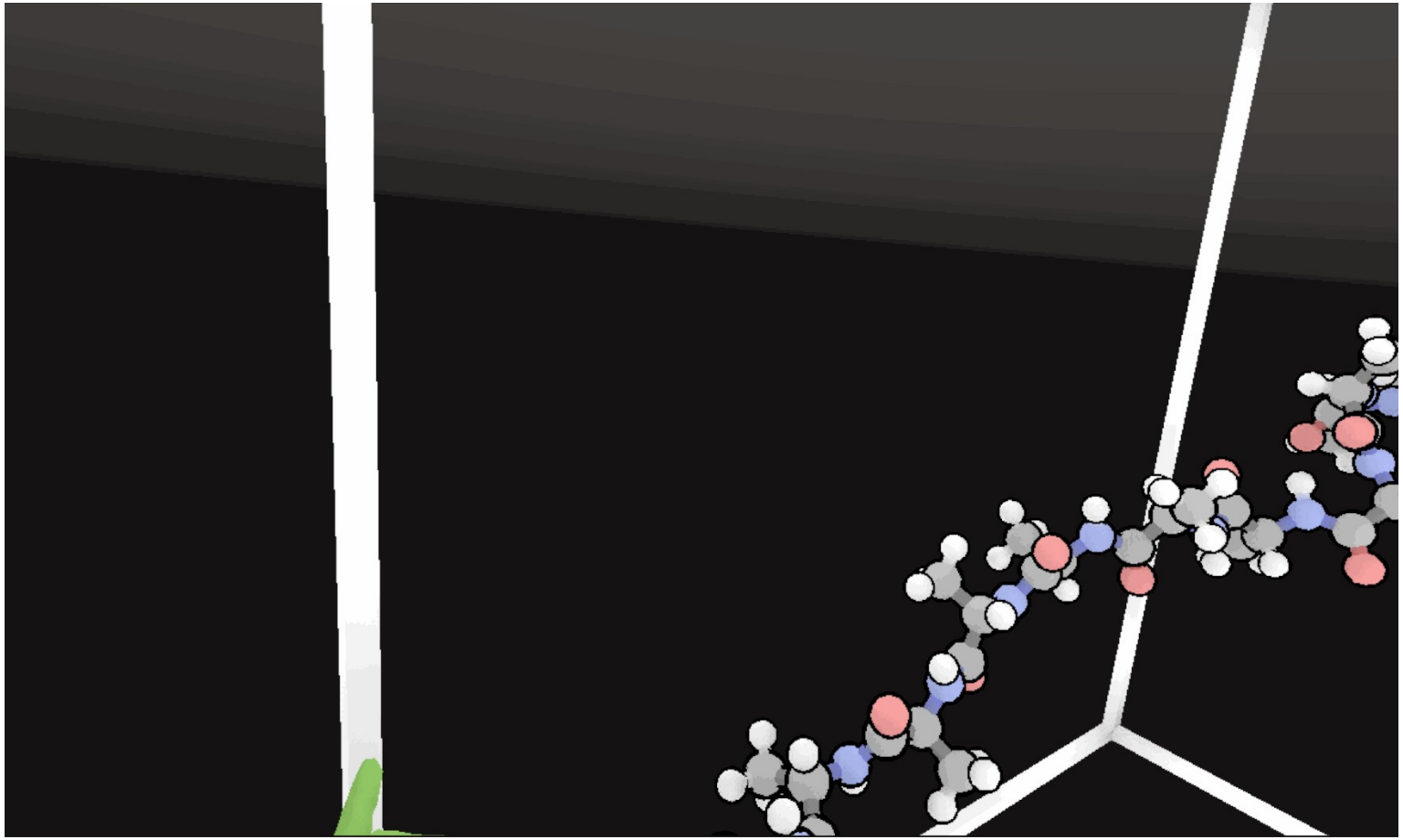
Garbage in, Garbage out.

- Classification very sensitive to rotations.
- Quaternions not the best features to use.
- Internal coordinates, or raw sensor data would be better.
- The gloves are a pain to calibrate!

Meanwhile, one student is clearly a Sith...



Tying a peptide in a knot



Conclusions

- It sort of works! But is also completely useless.
- TensorFlow estimators make it easy to get going.
- Also a good way to package up a model. Keras -> Estimator script exists.
- Can focus on features, the model, and hyperparameters.
- But Keras/scikit-learn have better support for pipelines/tuning.
- Fast-changing API makes trawling through the docs/issues difficult.
- Serving/exporting to other languages is a tricky maze to navigate.

Dear Hi5 Customers,

We have some new updates for UNITY including an interaction SDK. The new UNITY for Hi5 update includes correction of several crash issues with project files, and improvement of lagging issues.

UNITY INTERACTION SDK

The Hi5 UNITY Interaction SDK supports the interactions between hands and objects. Currently the functions include grabbing, pinching, releasing, throwing, lifting, poking, hand pressing and exchanging objects. More functionality will be added in future versions.

Thanks!