Early Stabilizing Feature Importance for TensorFlow Deep Neural Networks

IJCNN 2017, May 18, 2017

Jeff Heaton

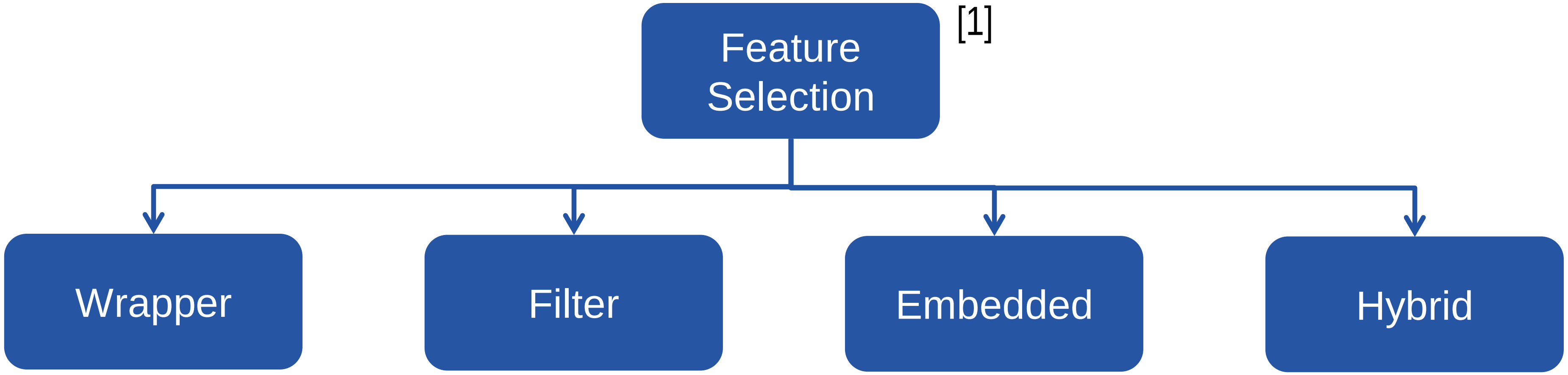
Steven McElwee

James Cannady

James Fraley

NOVA SOUTHEASTERN UNIVERSITY

This study addresses the problem that
there are not existing methods for feature importance ranking
that provide early stabilization or implementation
in Google TensorFlow deep neural networks.



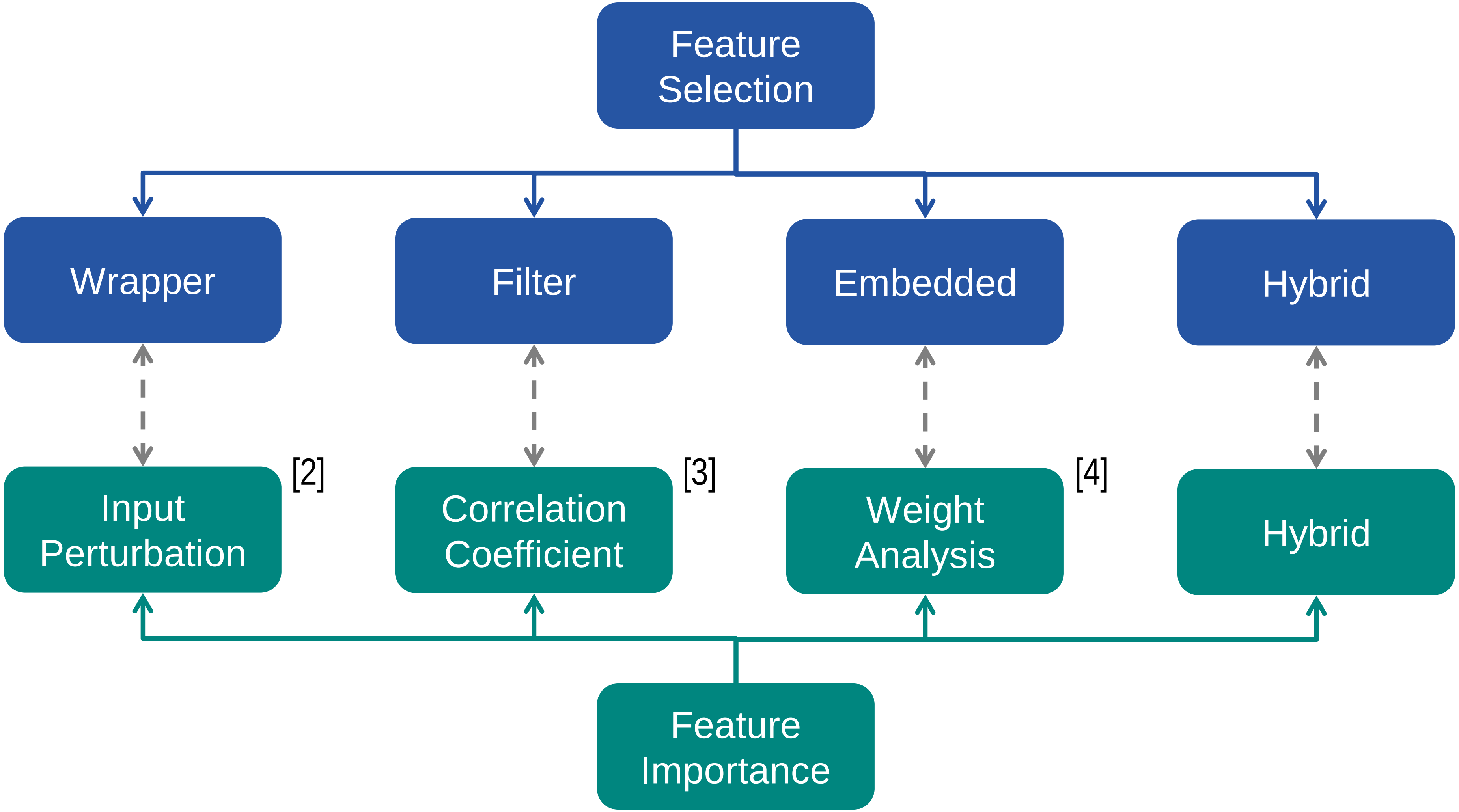
Wrapper: Dependent on model. Uses interaction of features.

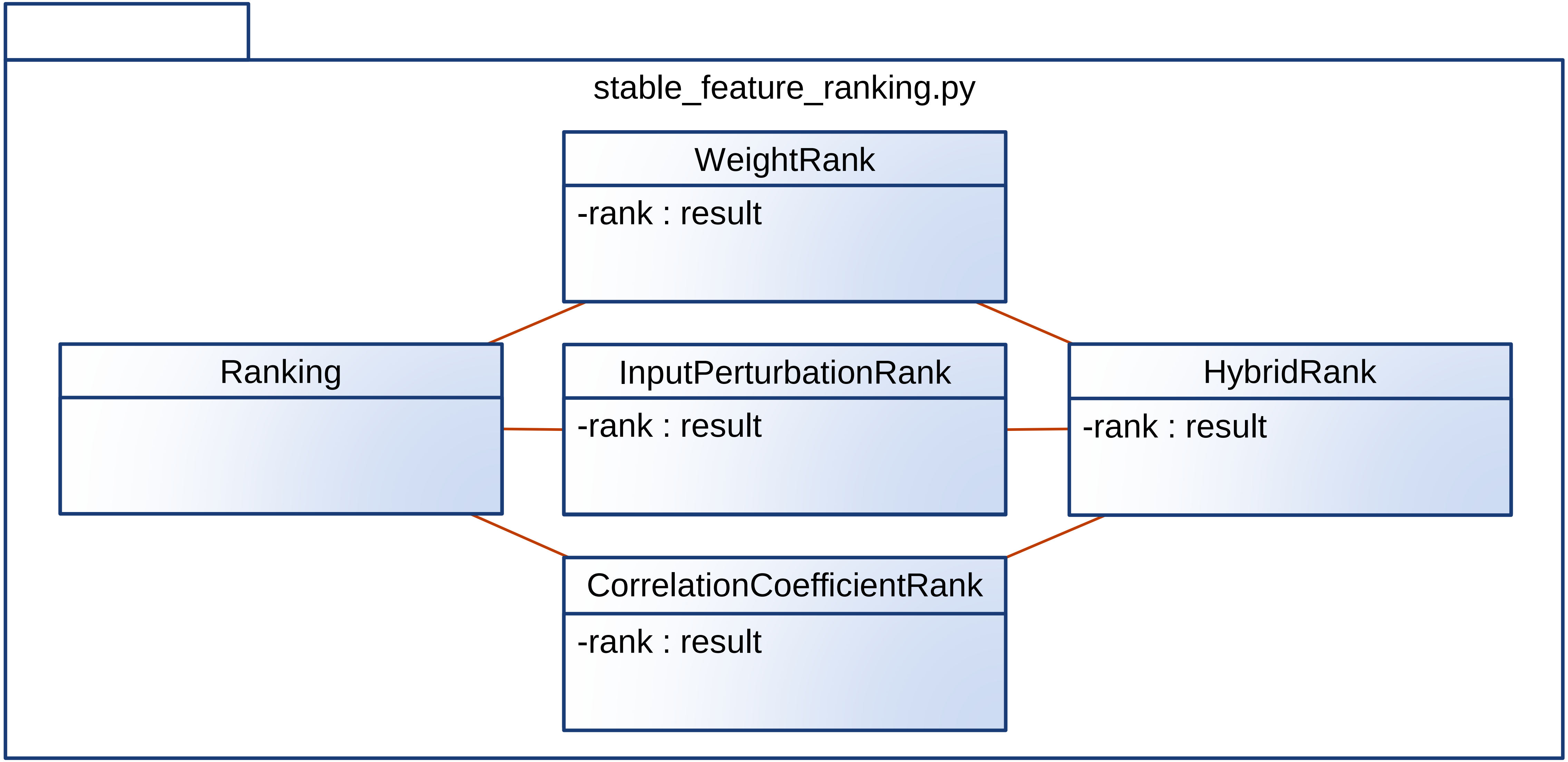
Filter: Independent of model. Ignores interaction of features.

Embedded: Built into learning algorithm.

Hybrid: Combines aspects of two or more feature selection approaches.

Relationship of Feature Selection & Importance





https://github.com/drcannady/Research/tree/master/projects/IJCNN-2017

Correlation Coefficient Feature Importance

- Pearson product-moment correlation coefficients
- Calculates each feature independently
- Strength: model independence
- Weakness: univariate analysis and feature redundancy

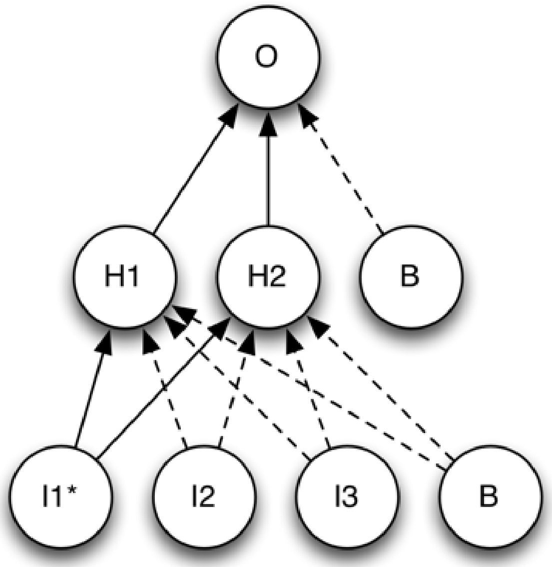
$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$$

Pseudocode:

```
function rank_stat(self, x, y):
  impt = []

  for i in range( numcols(x) ):
    c = corrcoef(x[: , i], y[: , 0])
    impt[i] = abs(c)

  impt = impt / sum(impt)

  return (impt)
```

Weight Analysis Feature Importance

- Relies on TensorFlow model
- Uses weights from inputs to first hidden layer
- Simplified version of Garson connection weight interpretation [21]



Pseudocode:

```
function rank_weight(x, y, network):
  weights = network.get_tensor_value(
    'dnn/layer0/Linear/Matrix:0')
  weights = weights ^ 2
  weights = sum(weights, axis=1)
  weights = sqrt(weights)
  weights = weights / sum(weights)
  return weights
```

$$a = \sqrt{\sum W^2}$$

Input Perturbation Feature Importance

- Shuffle input order and calculate MSE
- Wrong input values presented for each expected target
- Column maintains the same distribution
- No adverse effect on DNN other than the feature being perturbed
- Strength: no retraining needed
- Weakness: depends on the model

Pseudocode:

```
function rank_perturb(x, y, network):
  impt = dim(x.shape[1])

  for i in range(numcols(x)):
    hold = copy(x[, i])
    shuffle(x[, i])
    pred = network.predict(x)
    mse = mean_squared_error(y, pred)
    impt[i] = mse
    x[:, i] = hold
impt = impt / sum(impt)
return impt
```

Hybrid Feature Importance Algorithm Overview

Why Another Algorithm?

- Wrapper/Embedded Algorithms are usually accurate, but slow.
- Filter Algorithms are fast, but often least accurate.
- Wrapper/Embedded require a fully trained model for maximum accuracy.
- The feature importance ranking for Wrapper/Embedded will often change radically for a 25%, 50%, 75% and ultimately 100% trained neural network.
- This research sought an algorithm that stabilized the ranking early.

Algorithm Overview

- The hybrid algorithm uses:
  - Correlation Coefficient Rank
  - Input Perturbation
  - Weight Rank
- Hybrid algorithm uses the weight rank plus a weighted sum of input perturbation and correlation coefficient.
- The standard deviation of the normalized perturbation rank is used to balance input perturbation and the correlation coefficient rank.

Hybrid Feature Importance Technical Details

- Hybrid algorithm combines input perturbation, weight analysis, and correlation coefficient:

$$\boldsymbol{m} = \boldsymbol{w} + \boldsymbol{p}d + \boldsymbol{s}(1 - d)$$

- Perturbation rank weighted by:

$$d = \mathrm{SD}\left(\frac{\boldsymbol{p}}{\sum \boldsymbol{p}}\right)$$

- Correlation coefficient weighted by $1 - d$
- Requires fewer training iterations for large feature sets

Pseudocode:

```
function rank_hybrid(x, y, network):

  p_rank = rank_perturb(x, y, network)
  w_rank = rank_weight(network)
  s_rank = rank_stat(x, y)

  d = (np.std(p_rank / sum(p_rank)))

  impt = w_rank + (p_rank * d)
         + (s_rank * (1.0 - d))

  impt = impt / sum(impt)
```

Testing with Auto MPG Dataset

Perturbation Ranking Algorithm vs. Hybrid Ranking Algorithm

step	diff	1	2	3	4	5	6	7	8	9
0	0.13	yr	wt	dp	hp	o3	o2	ac	o1	cl
62	0.11	wt	yr	dp	hp	cl	o3	o1	ac	o2
124	0.09	wt	yr	dp	hp	o3	ac	o2	o1	cl
186	0.10	wt	yr	dp	hp	o1	ac	o3	o2	cl
248	0.09	wt	yr	dp	hp	cl	ac	o3	o2	o1
310	0.06	wt	yr	hp	dp	cl	o1	o3	o2	ac
372	0.05	wt	hp	yr	dp	o3	o1	ac	o2	cl
434	0.04	wt	hp	dp	yr	cl	o3	o1	ac	o2
496	0.02	wt	hp	yr	dp	cl	o1	ac	o3	o2
558	0.04	wt	hp	yr	dp	o1	o2	ac	o3	Cl
589	0.02	wt	hp	yr	dp	cl	o1	o3	ac	o2

steps	diff	1	2	3	4	5	6	7	8	9
0	0.07	wt	dp	hp	cl	yr	o1	o3	ac	o2
50	0.06	wt	dp	hp	cl	yr	o1	o3	ac	o2
100	0.06	wt	dp	hp	cl	yr	o1	o3	ac	o2
150	0.06	wt	dp	hp	cl	yr	o1	o3	ac	o2
200	0.05	wt	dp	hp	cl	yr	o1	o3	ac	o2
250	0.04	wt	dp	hp	cl	yr	o1	o3	ac	o2
300	0.04	wt	dp	hp	cl	yr	o1	o3	ac	o2
350	0.04	wt	dp	hp	cl	yr	o1	o3	ac	o2
400	0.02	wt	dp	hp	cl	yr	o1	o3	ac	o2
450	0.02	wt	dp	hp	cl	yr	o1	o3	ac	o2
475	0.03	wt	dp	hp	cl	yr	o1	o3	ac	o2

Novel Hybrid Ranking Algorithm stabilizes earlier than perturbation ranking algorithm.

Summary of Testing with Various Datasets and Algorithms

Data Set	Hybrid Steps	Perturbation Steps	Hybrid Diff	Correlation Diff
Auto MPG	25	475	0.05	0.09
Liver	6	114	0.08	0.23
WC Breast	13	247	0.08	0.08

Novel Hybrid Feature Importance Algorithm for Deep Neural Networks

Comparable feature importance to other established algorithms, but with earlier stabilization

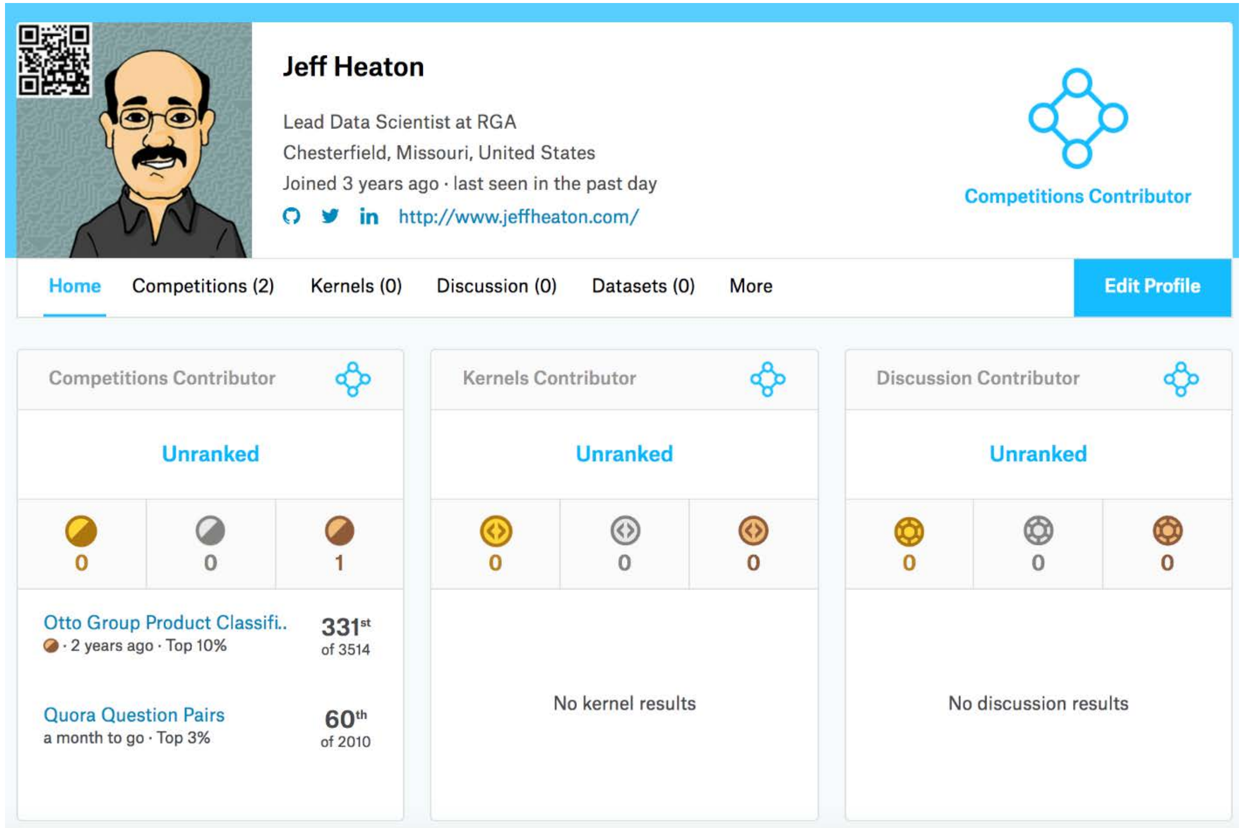
Feature Importance Toolkit for Google TensorFlow

Toolkit that implements the novel hybrid algorithm as well as correlation coefficient, input perturbation, and weight analysis algorithms

https://github.com/drcannady/Research/tree/master/projects/IJCNN-2017

Ongoing and Future Research

- We are using this algorithm as part of our submission for the Kaggle Quora question pairs challenge.
- Hybrid algorithm is used to quickly calculate the importance of over 20,000 N-Grams.
- This provides greater accuracy than TF-IDF.



References

[1] H. Lui & L. Yu. “Toward integrating feature selection algorithms for classification and clustering”. IEEE Transactions on Knowledge and Data Engineering, 17(4), 2005. 491-502..

[2] I. Guyon & A. Elisseeff. “An introduction to variable and feature selection”. Journal of Machine Learning Research, 3, 1, 2003. 157-1,182.

[3] F. Ahmad, N. Norwawi, S. Deris & N. Othman. “A review of feature selection techniques via gene expression profiles”. Proceedings of the International Symposium on Information Technology, 2, 2008. 1-7.

[4] G. Garson. “Interpreting neural-network connection weights”. Artificial Intelligence Expert 6, 1991. 47–51.