Vowpal Wabbit

http://hunch.net/~vw/

git clone
git://github.com/JohnLangford/vowpal_wabbit.git

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.
2. Hashing. Raw text is fine. A valid input is:
   1 | The dog ate my homework

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.
2. Hashing. Raw text is fine. A valid input is:
   1 | The dog ate my homework
3. Allreduce. Terascale learning paper = most scalable public algorithm.

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.
2. Hashing. Raw text is fine. A valid input is:
   1 | The dog ate my homework
3. Allreduce. Terascale learning paper = most scalable public algorithm.
4. Reductions. Solve wide variety of problems well by reduction to simple problems.

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.
2. Hashing. Raw text is fine. A valid input is:
   1 | The dog ate my homework
3. Allreduce. Terascale learning paper = most scalable public algorithm.
4. Reductions. Solve wide variety of problems well by reduction to simple problems.
5. Interactive. Causation instead of correlation. Learn to control based on feedback.

Vowpal Wabbit: Research Delivery Vehicle

1. Online by default. A good solution to an ML problem is always an hour or less away.
2. Hashing. Raw text is fine. A valid input is:
   1 | The dog ate my homework
3. Allreduce. Terascale learning paper = most scalable public algorithm.
4. Reductions. Solve wide variety of problems well by reduction to simple problems.
5. Interactive. Causation instead of correlation. Learn to control based on feedback.
6. Learn2Search. See Hal/Kai-Wei later
7. others....

A user base becomes addictive

1. Mailing list of >400

A user base becomes addictive

1. Mailing list of >400
2. The official strawman for large scale logistic regression (see demos/booths) :-)

A user base becomes addictive

1. Mailing list of >400
2. The official strawman for large scale logistic regression (see demos/booths) :-)
3.

An example

`vw -c rcv1.train.raw.txt -b 22 --ngram 2 --skips 4 -l 0.25 --binary` provides stellar performance in 12 seconds.

Improved: Learning Reductions

The core idea: reduce complex problem A to simpler problem B then use solution on B to get solution on A.

Problems:

1. How do you make it efficient enough?
2. How do you make it natural to program?

The Reductions Interface

```
void learn(learner& base, example& ec)
{
   base.learn(ec); // The recursive call
   if ( ec.pred.scalar > 0) //Thresholding
      ec.pred.scalar = 1;
   else
      ec.pred.scalar = -1;
   if (ec.l.simple.label == ec.pred.scalar)
      ec.loss = 0.; // Loss Definition
   else
      ec.loss = ec.l.simple.weight;
}
```

New: Logarithmic Online Multiclass Tree

See: http://arxiv.org/abs/1406.1822

Repeatedly:

1. See x
2. Predict $\hat{y} \in \{1, ..., K\}$
3. See y

New: Logarithmic Online Multiclass Tree

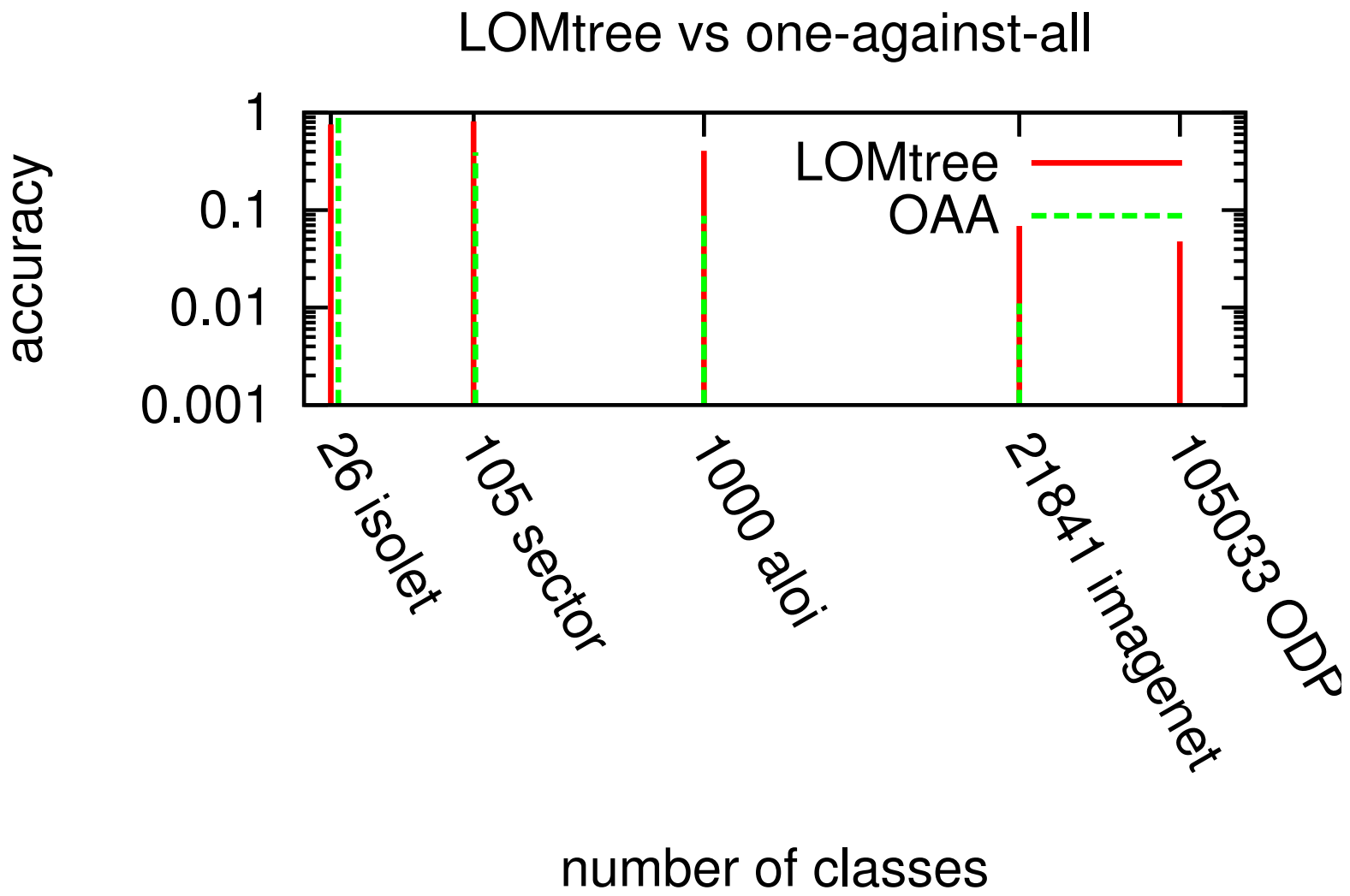
See: http://arxiv.org/abs/1406.1822

Repeatedly:

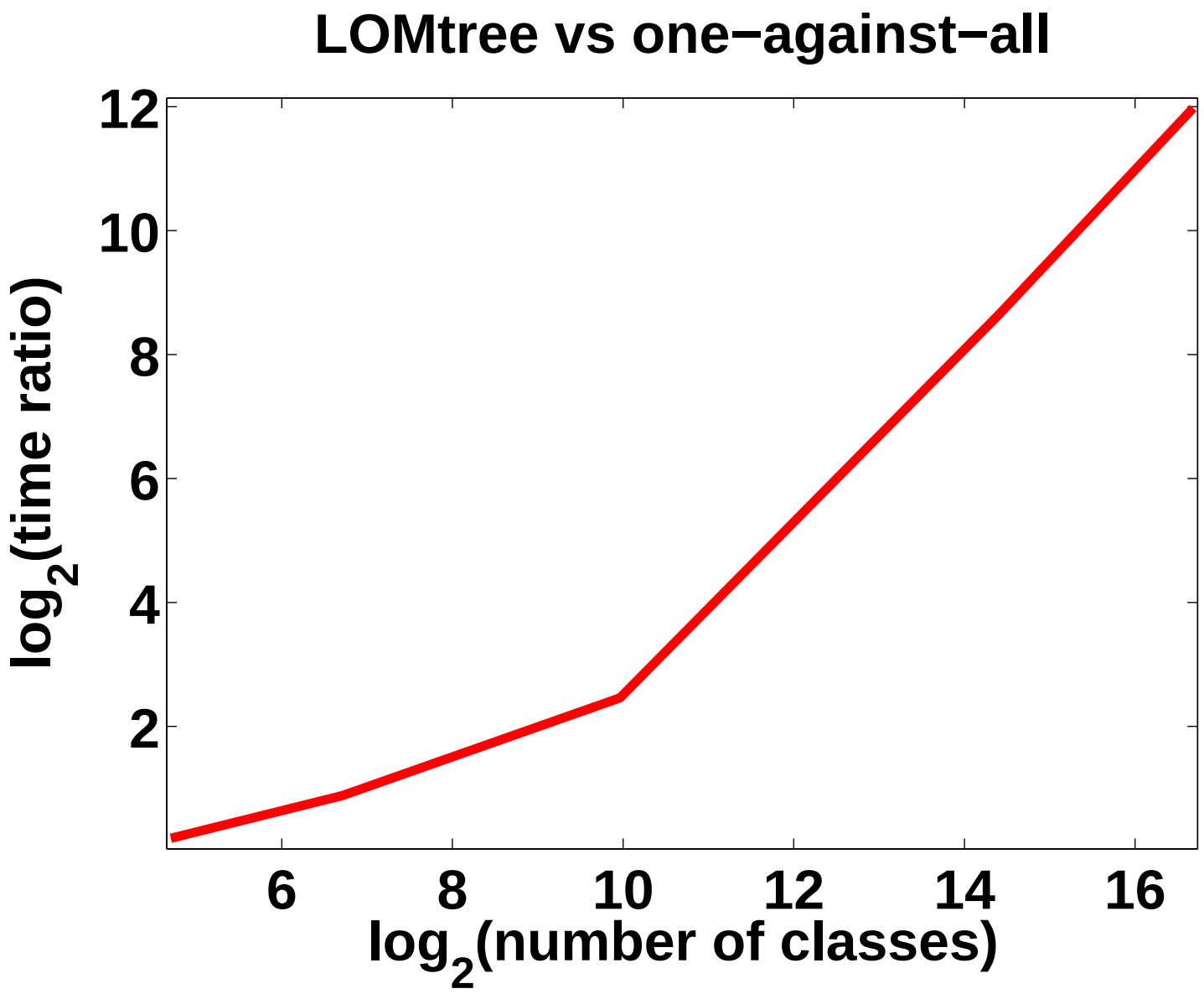
1. See x
2. Predict $\hat{y} \in \{1, ..., K\}$
3. See y

Goal: Goal in $O(\log K)$ time minimize loss.

Accuracy for a fixed training time



Classes vs Test time ratio



The Method

A dynamically created tree with classifiers in the nodes and multiple leaves per label operating by reduction. ~550 LOC.

Usage:

```
vw --log_multi <nodes> <dataset>
```

New: The Exploration Library

(With: Luong Hoang, Sid Sen, Sarah Bird)

NIPS tutorial last year: Learning to interact without exploration is bogus.

New: The Exploration Library

(With: Luong Hoang, Sid Sen, Sarah Bird)

NIPS tutorial last year: Learning to interact without exploration is bogus.
Problem: Exploration must happen deep inside existing systems.

New: The Exploration Library

(With: Luong Hoang, Sid Sen, Sarah Bird)

NIPS tutorial last year: Learning to interact without exploration is bogus.
Problem: Exploration must happen deep inside existing systems.
Solution: An exploration library which randomizes correctly & helps log the right things.

New: VW in AzureML

(see: http://tinyurl.com/vw-azureml)

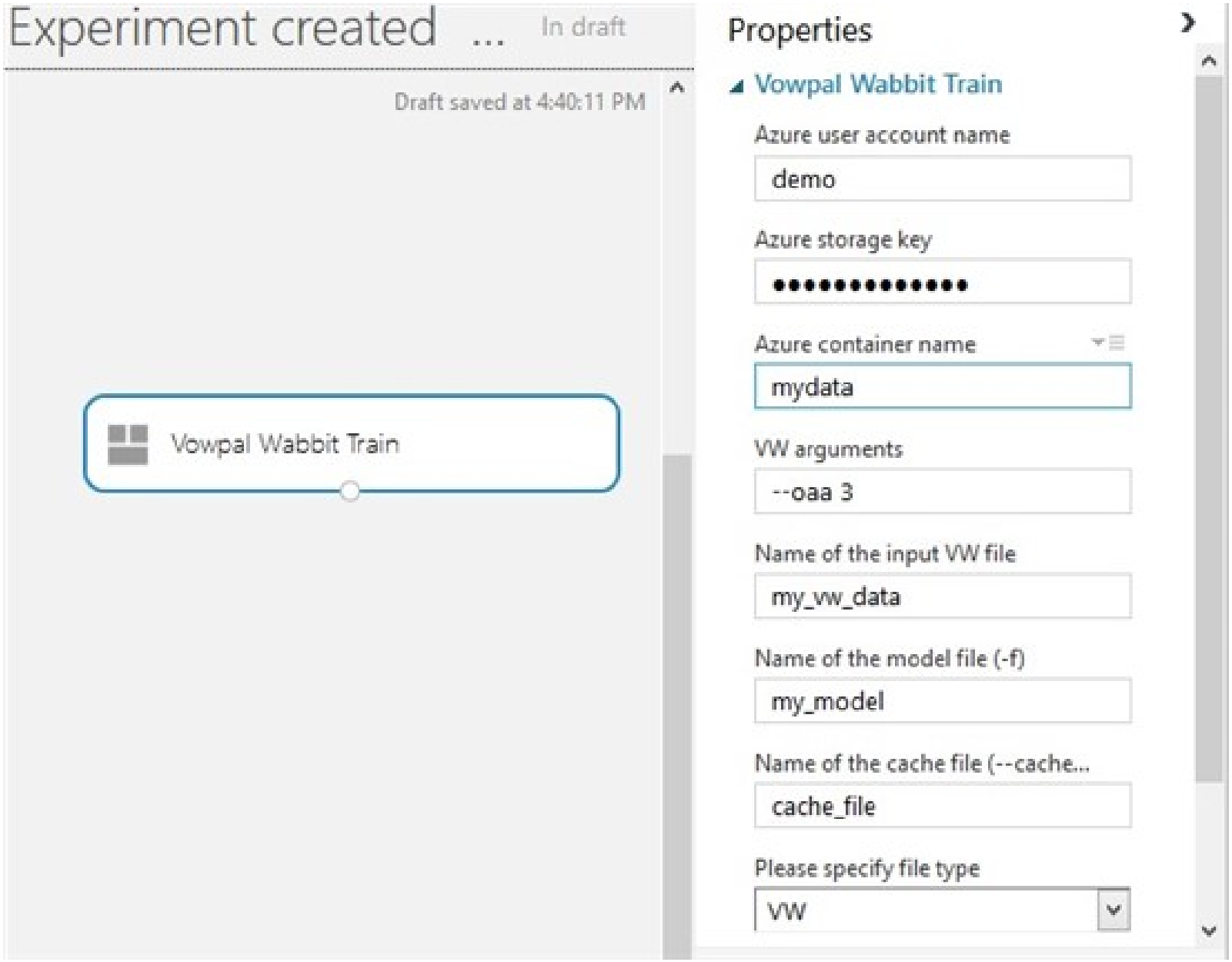
Problem: You want to deploy a model for large scale use.

New: VW in AzureML

(see: http://tinyurl.com/vw-azureml)

Problem: You want to deploy a model for large scale use.
Solution: Azure ML cloud service.

It even has browser GUI



The plan

1. John: Intro++
2. Matus: Polynomial learning
3. Paul: LRQ + Hogwild mode
4. Hal: Learning 2 search + Python
5. Kai-Wei: Entity Relation + Dependency Parsing
6. Alekh: Online SVM