CSAIL

MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

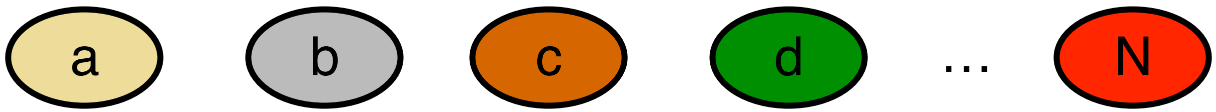
Limited Labels for Unlimited Data: Active Learning for Speaker Recognition

Stephen Shum, Najim Dehak, Jim Glass

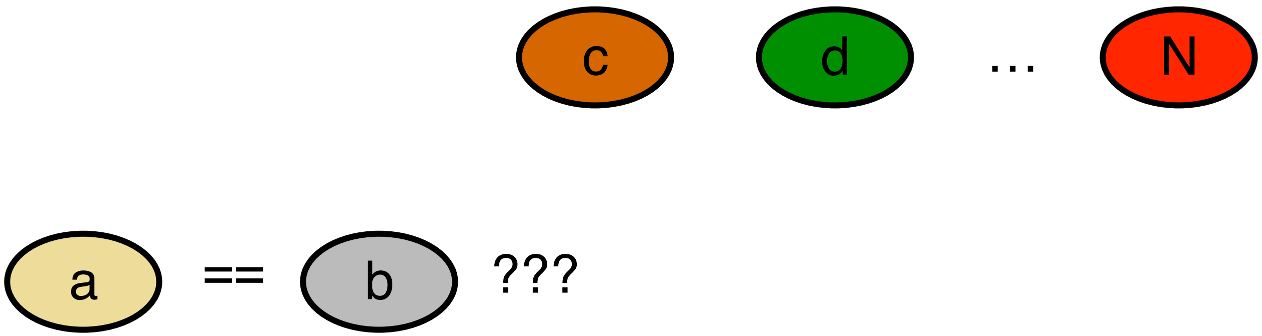
September 15, 2014

How much labeled data do we *really* need to build a state-of-the-art speaker recognition system?

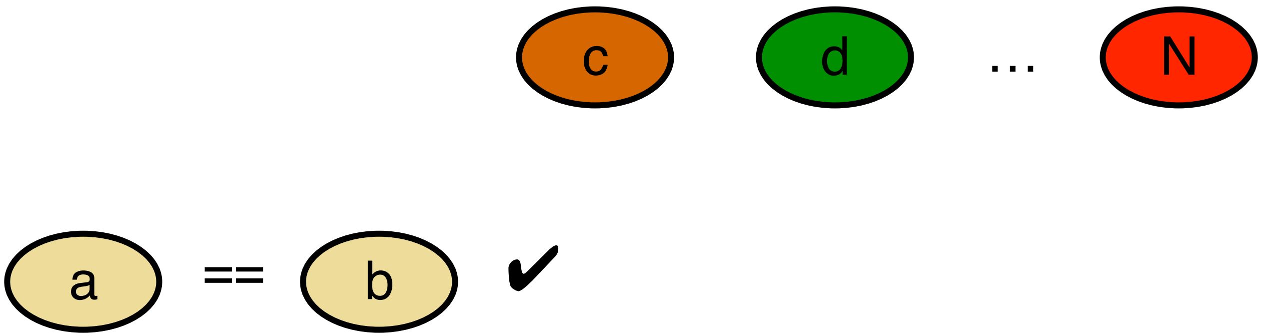
N unlabeled utterances



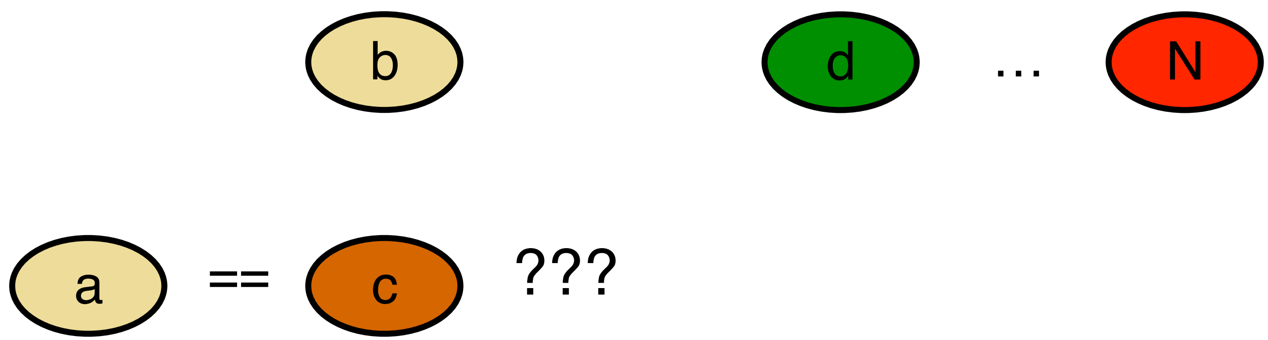
N unlabeled utterances



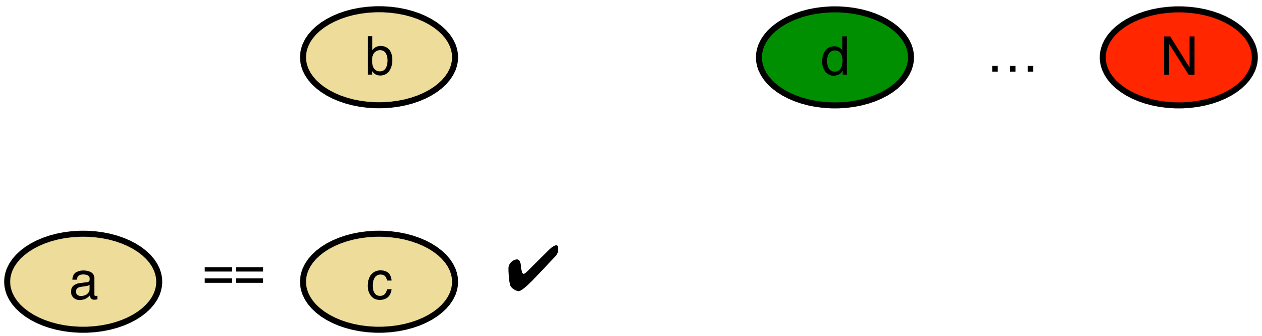
N unlabeled utterances



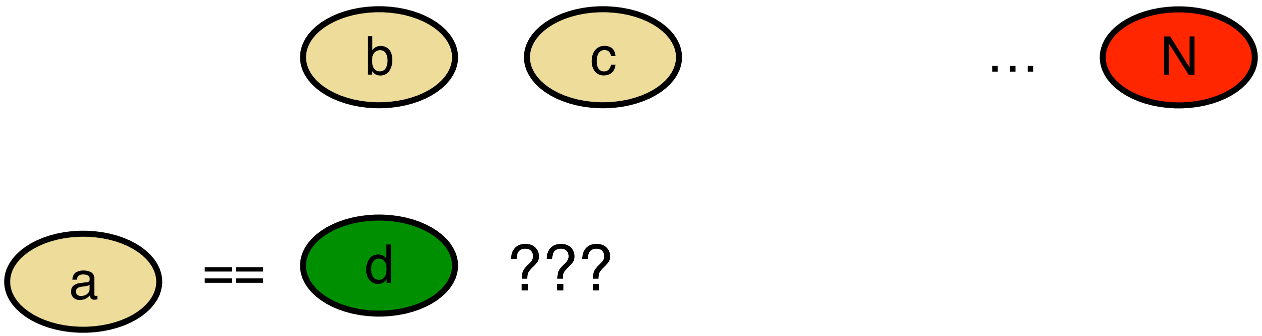
N unlabeled utterances



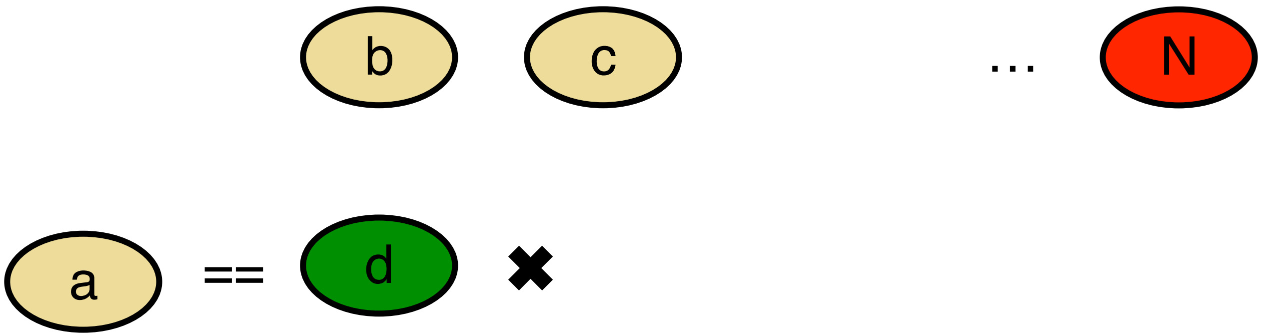
N unlabeled utterances



N unlabeled utterances

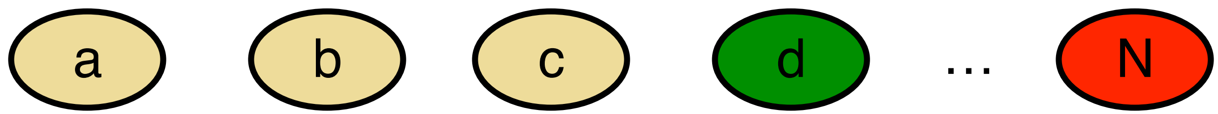


N unlabeled utterances



N unlabeled utterances

- O(N^2) queries is expensive!



Problem Statement

- **Lots of unlabeled utterances**
  – NIST 2004, 2005, 2006, 2008 Speaker Recognition Evaluations (SRE)
- **Evaluate on 2010 NIST SRE**

- **Similar to previous work on domain adaptation**
    * **Aronowitz, 2014; Brummer, 2014; Garcia-Romero, 2014; Glembek, 2014; Shum, 2014; et cetera**
  – Here, **NO** previously labeled data is allowed

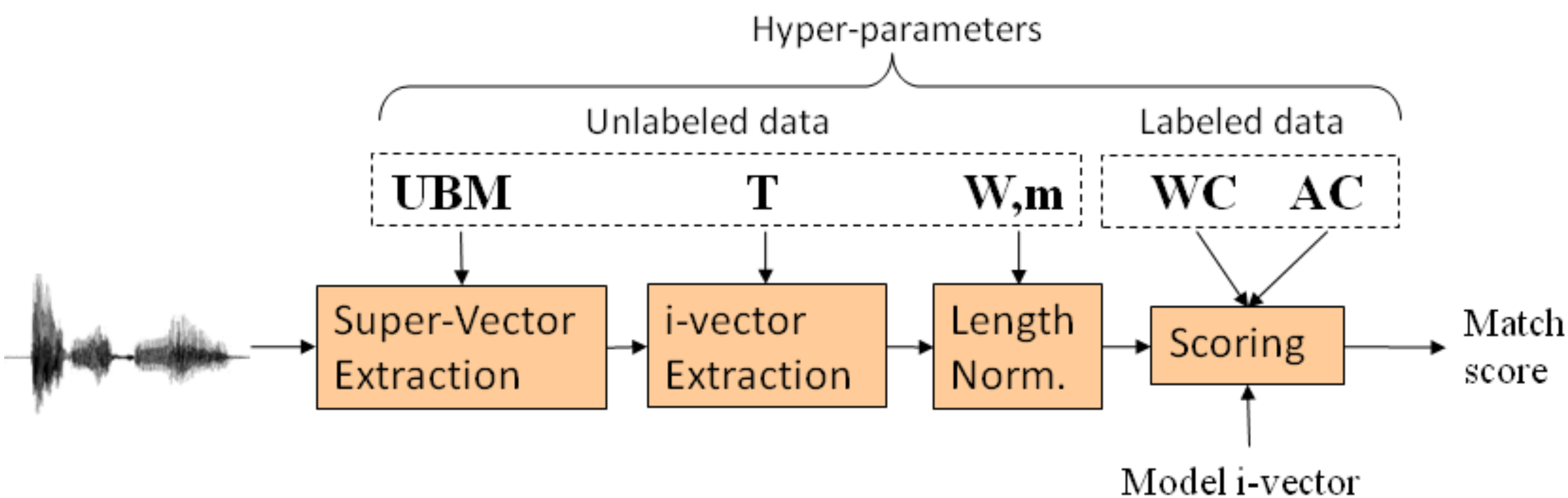
- **Allow pairwise queries to some noiseless oracle**
  – “Do utterances A and B contain the same speaker?”

Problem Statement

- **Objectives**
  - Minimize number of pairwise queries
  - Maximize performance on speaker recognition
- **Take-away**
  - The actual number of pairwise labels needed to obtain state-of-the-art results is a mere fraction of the queries needed to exhaustively label an entire set of utterances from scratch.

Experiment Setup

- 600-dimensional i-vectors
- Gender-independent UBM (2048 Gaussians)

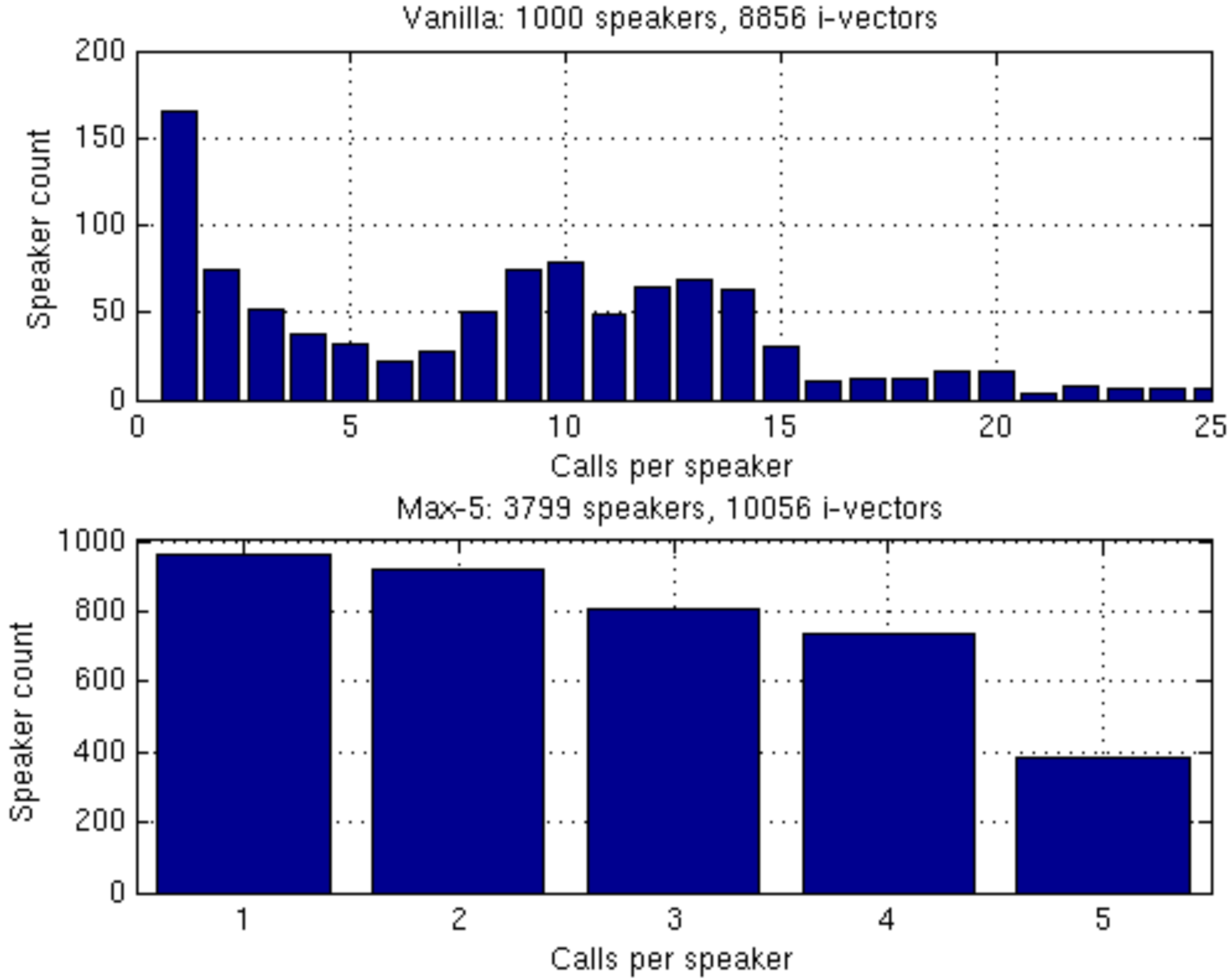


Sampling from the NIST Data

- **3800 unique speakers**
  - 1100 male, 2700 female
- **33,000 phone calls**
  - Calls per speaker = 8.7
  - Phone numbers per speaker = 2.8

- **Sampled subsets from the data**
  - Lets us explore how performance might vary under datasets that have different distributions of utterances per speaker

Sampling from the NIST Data



Roadmap

- **Motivation**
- **Problem Statement**
- **Experiment Setup**
  - Sampling from the NIST data
- **Algorithm**
  - Practical implementation details
  - Other design choices
- **Results**
- **Discussion**

Roadmap

- Motivation
- Problem Statement
- Experiment Setup
  - Sampling from the NIST data
- Algorithm
  - Practical implementation details
  - Other design choices
- Results
- Discussion

Algorithm

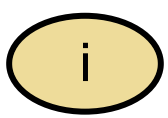
- **Compare between i-vectors via the cosine similarity**

- **Graph terminology**
  - Each utterance (or i-vector) is represented as a **node**
  - Connect two i-vectors with an **edge** if they are from the same speaker

- **Initialization**
  - Completely disconnected graph (i.e., no edges!)

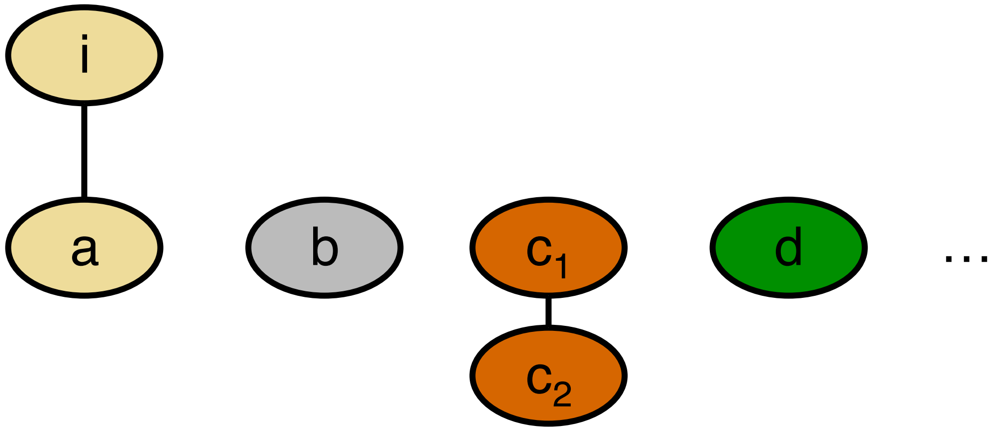
Algorithm

- **Pick an i-vector, i.**



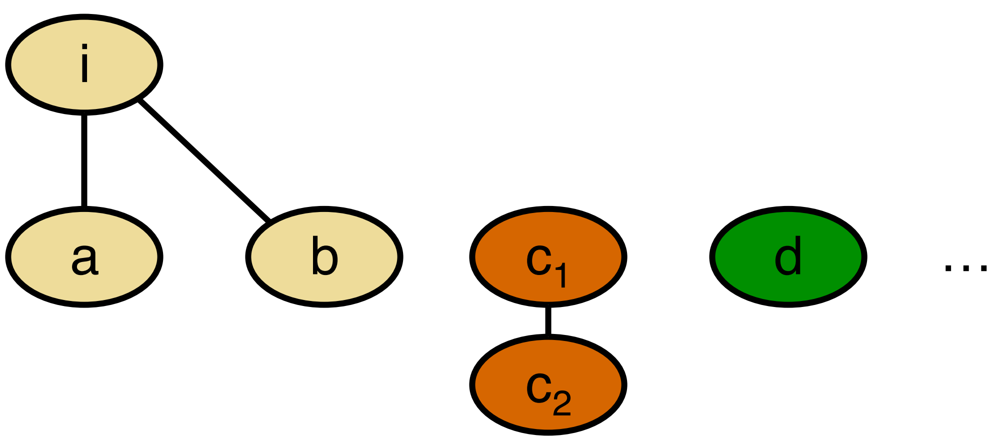
Algorithm

- Pick an i-vector, i.
- Query i against its neighbors in order of decreasing cosine similarity.



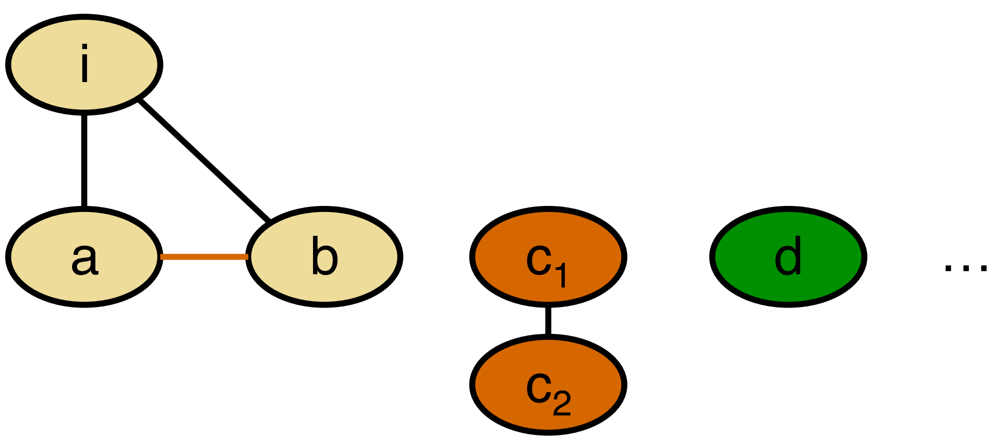
Algorithm

- Pick an i-vector, i.
- Query i against its neighbors in order of decreasing cosine similarity.



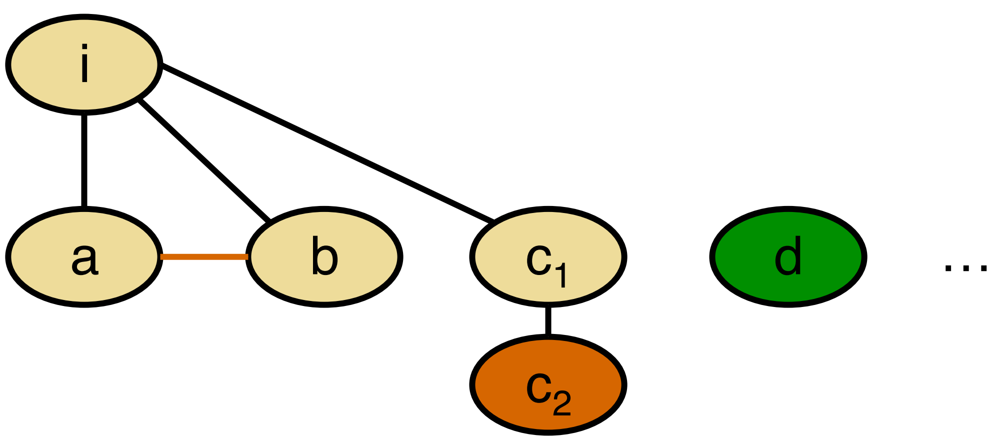
Algorithm

- **Pick an i-vector, i.**
- **Query i against its neighbors in order of decreasing cosine similarity.**
  - Automatically turn all "same" pairs into fully connected cliques.



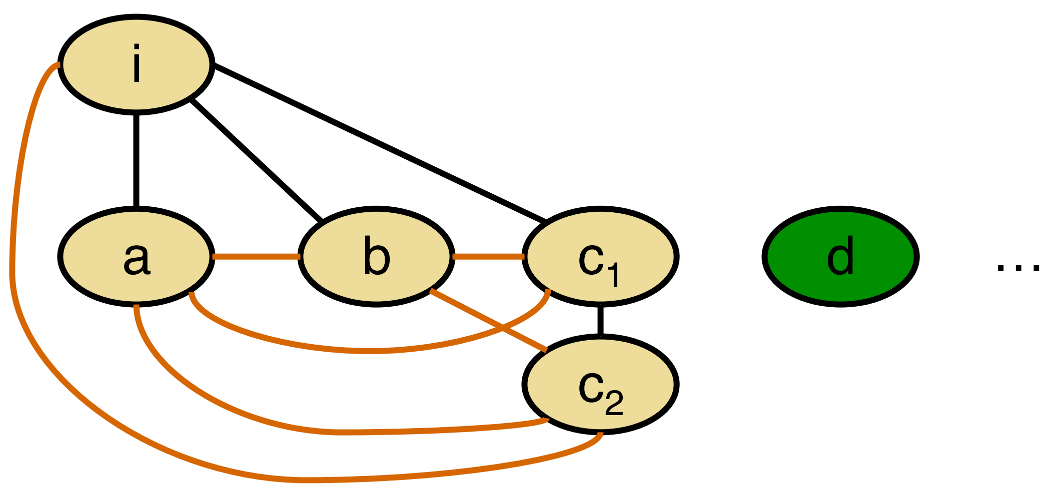
Algorithm

- **Pick an i-vector, i.**
- **Query i against its neighbors in order of decreasing cosine similarity.**
  - Automatically turn all “same” pairs into fully connected cliques.



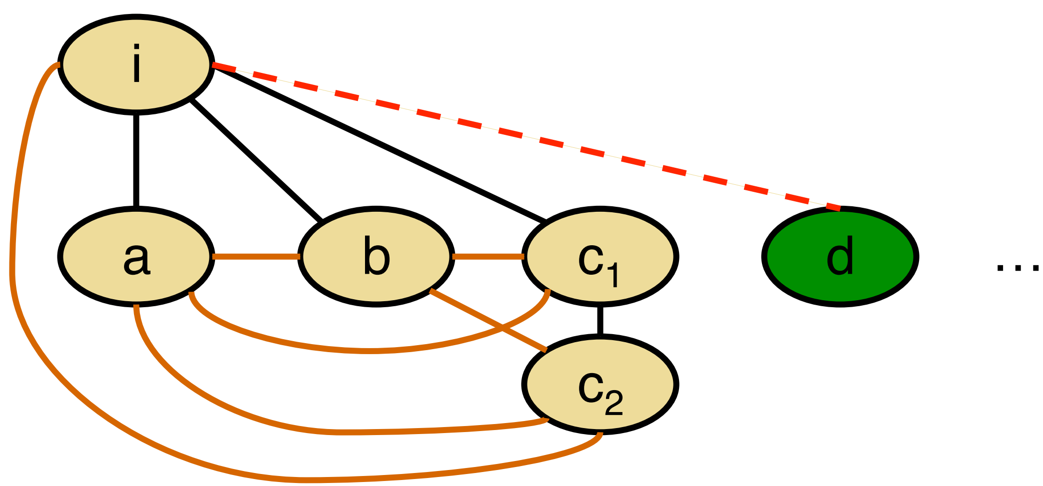
Algorithm

- **Pick an i-vector, i.**
- **Query i against its neighbors in order of decreasing cosine similarity.**
  - Automatically turn all “same” pairs into fully connected cliques.



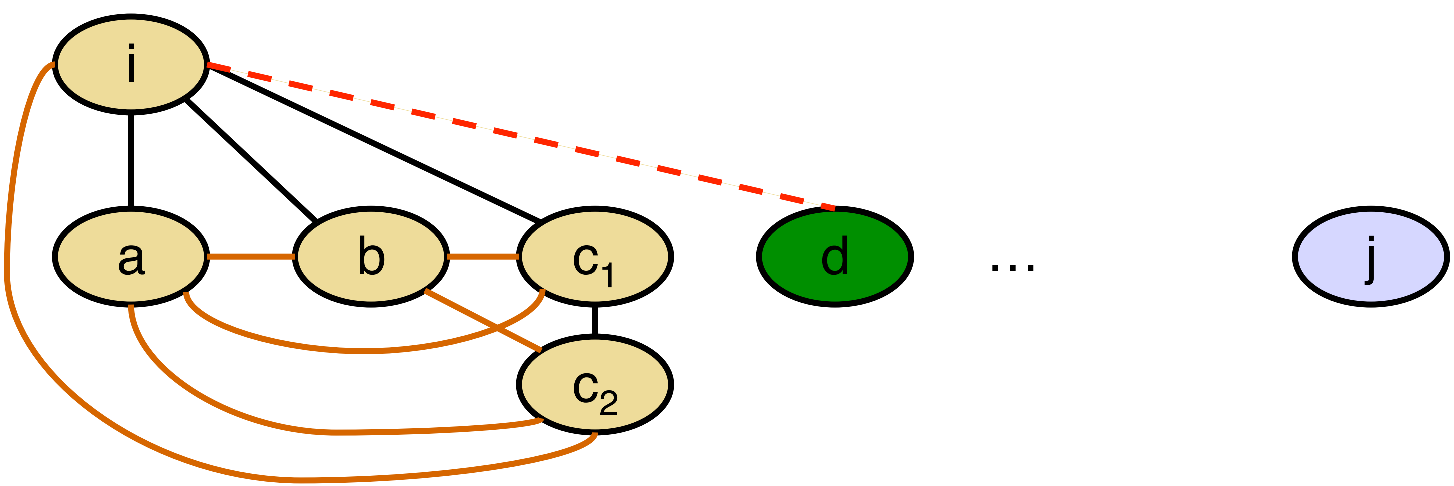
Algorithm

- Pick an i-vector, i.
- Query i against its neighbors in order of decreasing cosine similarity.
  - Automatically turn all “same” pairs into fully connected cliques.
- Stop when oracle returns “different” for some pair (i, d).



Algorithm

- Pick an i-vector, i.
- Query i against its neighbors in order of decreasing cosine similarity.
  - Automatically turn all “same” pairs into fully connected cliques.
- Stop when oracle returns “different” for some pair (i, d).
- Pick another i-vector that is as far away as possible.



Practical Implementation

- **All pairwise cosine similarities → affinity matrix**
  - Single matrix multiplication
- **Finding neighbors to query**
  - Sort each row of the affinity matrix
- **Finding an i-vector that is as “far away” as possible**
  - Average relevant rows of the affinity matrix and pick the index corresponding to the minimal value

Some Other Design Choices

- **Just presented the "greedy coverage" approach**

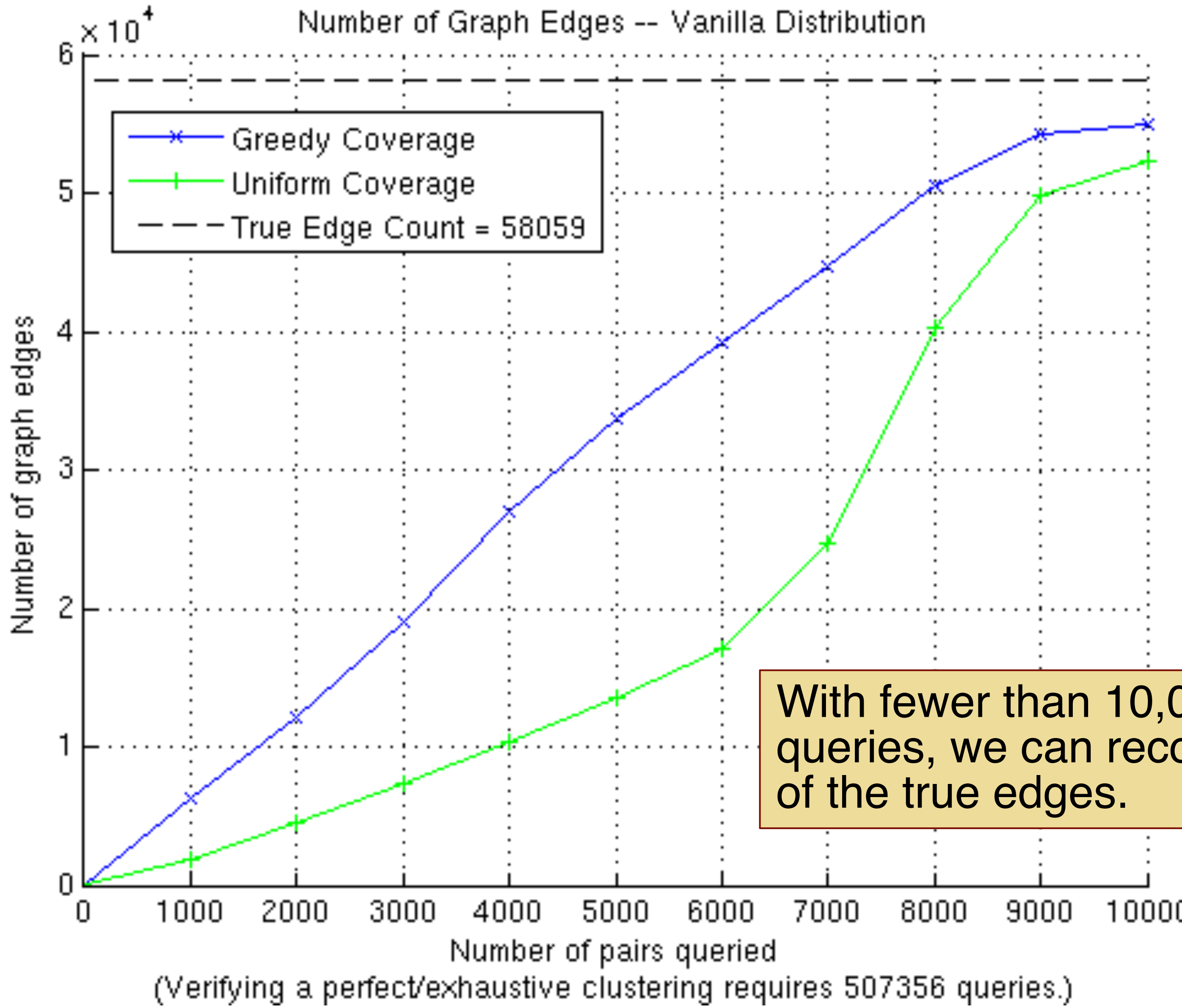
- **Experiments compare against "uniform coverage" approach**
  - Query every unique i-vector's 1st nearest neighbor, then 2nd, and so on
  - Every i-vector is considered at least once every N queries
  - Slow to obtain reasonable estimate of speaker within-class variability

- **Also tried "global score sort"**
  - Pool together all similarity scores, globally
  - Query individual pairs in order of decreasing score
  - Higher similarity scores indicated denser neighborhoods of i-vectors, not necessarily regions of strong within-speaker similarity

Roadmap

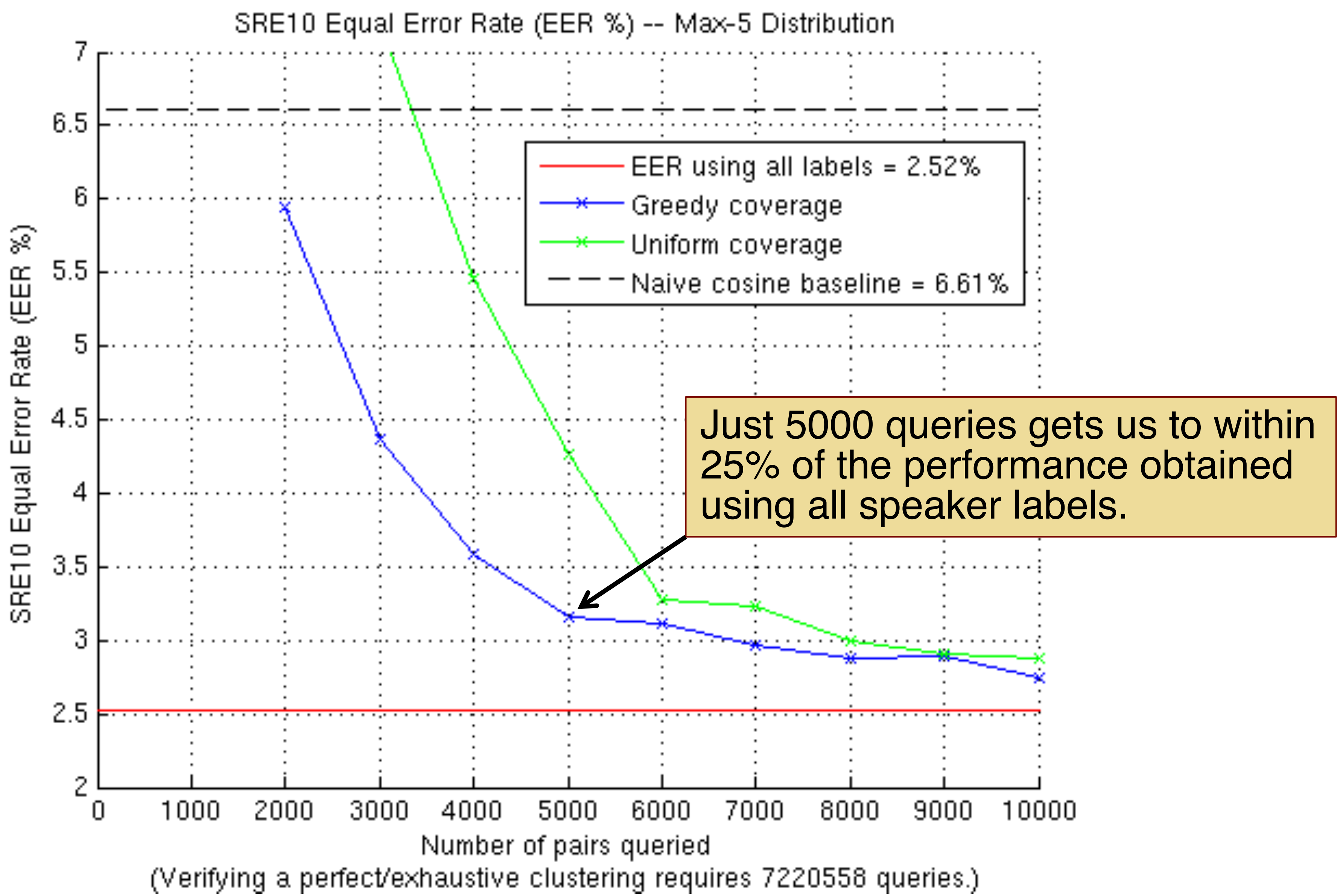
- Motivation
- Problem Statement
- Experiment Setup
  - Sampling from the NIST data
- Algorithm
  - Practical implementation details
  - Other design choices
- Results
- Discussion

Graph Edges vs. Pairs Queried

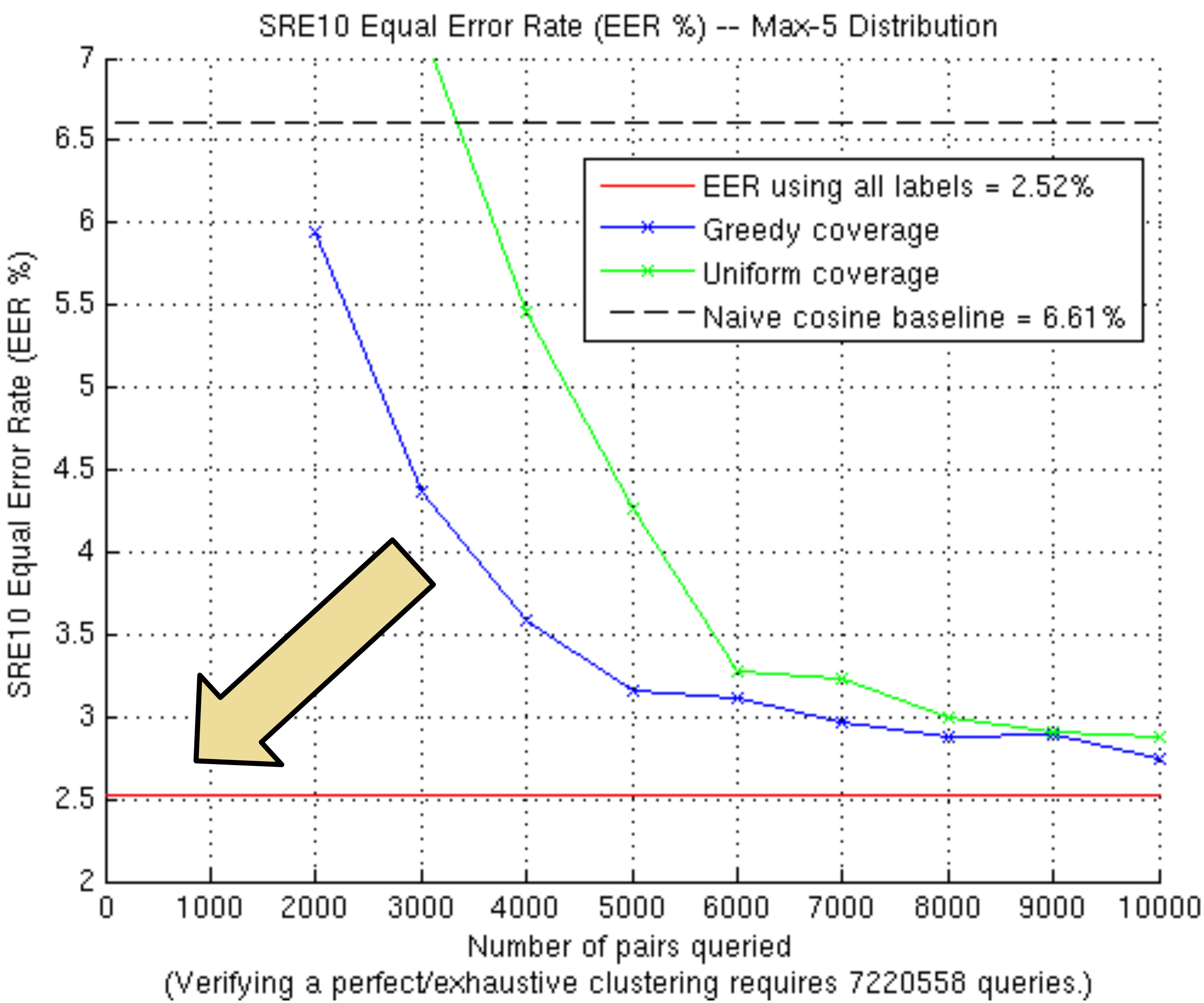


With fewer than 10,000 pairwise queries, we can recover over 90% of the true edges.

Speaker Recognition Performance



Speaker Recognition Performance



Ongoing Investigations

- **Data re-representation**
  - Key element in active learning
- **Incorporating prior knowledge**
  - Domain adaptation challenge gave us labels to Switchboard data
- **Extrapolating labels via semi-supervised clustering**
- **Noisy labels**
  - A noiseless oracle is a big assumption!
  - Humans, both expert and naïve listeners, are not perfect (Shen, 2011).

Conclusion

- **Attempted to quantify the amount of labeled data needed to build a speaker recognition system.**
  - The actual number of pairwise labels needed to obtain state-of-the-art results is a mere fraction of the queries required to fully label an entire set of utterances.

- **What are other ways in which we can leverage the power of pairwise comparisons?**
  - "Do utterances A and B contain the same ___________ ?"