CSCI567 Machine Learning (Spring 2018)

Michael Shindler

Lecture 20: April 2 2018

Outline

Outline

Quiz 2 coming up

- It is now week 12
- Friday April 6 is coming up. Quiz 2.
- Quiz 2:
  - **DO NOT OPEN EXAM UNTIL TOLD TO DO SO**
  - Bring a pencil
  - Bring your USC ID.
  - Be sure to fill out the ID section on Scantron
  - Be sure to STOP when time called
    - Stop writing immediately.
    - Look up, not at your exam or desk.
  - Know your name, ID#, lecture room, enrolled discussion time

Outline

Markov chain

Definition

Given a sequentially ordered random variables $X_1, X_2, \cdots, X_t, \cdots, X_T$, called *states*,

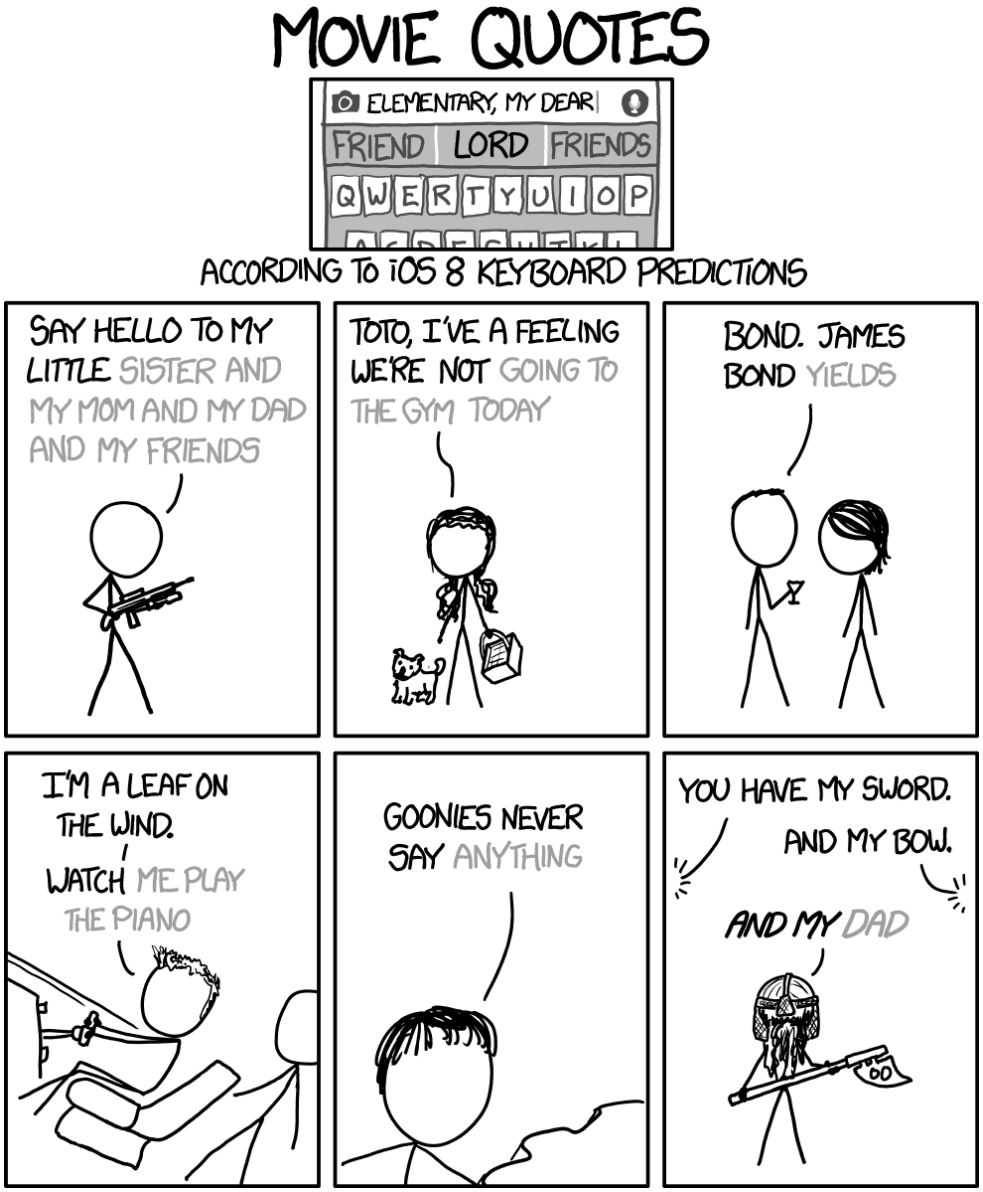
- **Transition probability** for describing how the state at time $t-1$ changes to the state at time t,

$$P(X_t = \text{value}' | X_{t-1} = \text{value})$$

- **Initial probability** for describing the initial state at time $t = 1$.

$$P(X_1 = \text{value})$$

value represents possible values $\{X_t\}$ can take. Note that we will assume that all the random variables (at different times) can take value from the same set and assume that the transition probability does not change with respect to time t, i.e., a stationary Markov chain.


MOVIE QUOTES
ELEMENTARY, MY DEAR
FRIEND LORD FRIENDS
QWERTYUIOP
ACCORDING TO iOS 8 KEYBOARD PREDICTIONS
SAY HELLO TO MY LITTLE SISTER AND MY MOM AND MY DAD AND MY FRIENDS
TOTO, I'VE A FEELING WE'RE NOT GOING TO THE GYM TODAY
BOND. JAMES BOND YIELDS
I'M A LEAF ON THE WIND. WATCH ME PLAY THE PIANO
GOONIES NEVER SAY ANYTHING
YOU HAVE MY SWORD. AND MY BOW. AND MY DAD

Maximum likelihood estimation

$$\sum_m \log P(\boldsymbol{x}^m) = \sum_m \log P(x_1^m) + \sum_m \sum_t \log P(x_t^m | x_{t-1}^m)$$
$$= \sum_m \log \pi_{x_1^m} + \sum_m \sum_t \log a_{x_{t-1}^m \, x_t^m}$$

Maximizing this, we will get (derivation is left as an exercise)

$$\pi_i = \frac{\#\text{of sequences starting with } i}{\#\text{of sequences}}$$

and

$$a_{ij} = \frac{\#\text{of transitions starting with } i \text{ but ending with } j}{\#\text{of transitions starting with } i}$$

Example

Suppose we have two possible states $X_t \in \{0, 1\}$, and we have observed the following 3 sequences

$$1\ 0\ 0\ 1$$
$$0\ 1\ 1\ 1$$
$$1\ 1\ 1\ 1$$

Thus

$$\pi_0 = \frac{1}{3}, \quad \pi_1 = \frac{2}{3}$$

and

$$a_{00} = \frac{1}{3}, \quad a_{01} = \frac{2}{3}$$
$$a_{10} = \frac{2}{6}, \quad a_{11} = \frac{4}{6}$$

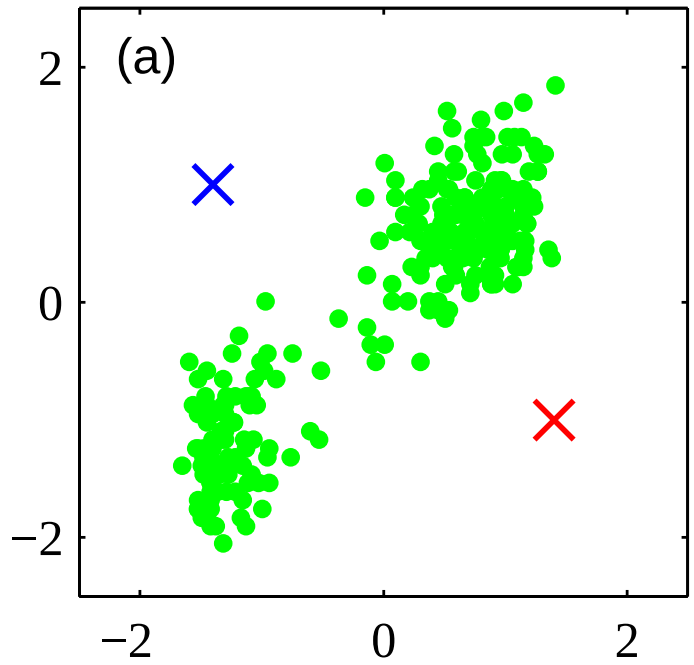
Outline

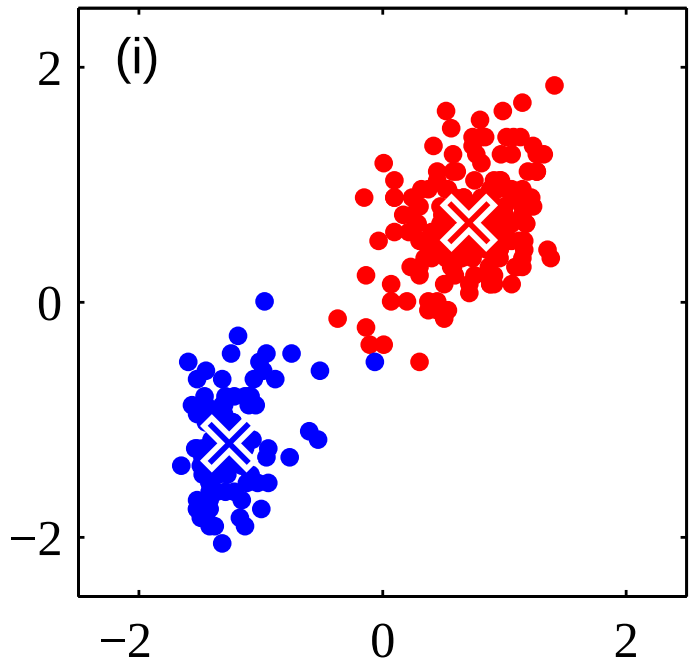
Clustering

Setup Given $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ and K, we want to output

- $\{\boldsymbol{\mu}_k\}_{k=1}^K$: prototypes of clusters
- $A(\boldsymbol{x}_n) \in \{1, 2, \ldots, K\}$: the cluster membership, i.e., the cluster ID assigned to $\boldsymbol{x}_n$

Example Cluster data into two clusters.





Clustering

Setup Given $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ and K, we want to output

- $\{\boldsymbol{\mu}_k\}_{k=1}^K$: prototypes of clusters
- $A(\boldsymbol{x}_n) \in \{1, 2, \ldots, K\}$: the cluster membership, i.e., the cluster ID assigned to $\boldsymbol{x}_n$

Key difference from *supervised learning problems*

Nobody tells us what the ground-truth is for any $\boldsymbol{x}_n$!

Algorithm: K-means clustering

Intuition Data points assigned to cluster k should be close to $\boldsymbol{\mu}_k$, the prototype.

Algorithm: K-means clustering

Intuition Data points assigned to cluster k should be close to $\boldsymbol{\mu}_k$, the prototype.

Distortion measure (clustering objective function, cost function)

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

where $r_{nk} \in \{0, 1\}$ is an indicator variable

$$r_{nk} = 1 \quad \text{if and only if} \quad A(\boldsymbol{x}_n) = k$$

Lloyd's Algorithm for k-means Clustering

Minimize distortion measure alternative optimization between $\{r_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$

- **Step 0** Initialize $\{\boldsymbol{\mu}_k\}$ to some values

Lloyd's Algorithm for k-means Clustering

Minimize distortion measure alternative optimization between $\{r_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$

- **Step 0** Initialize $\{\boldsymbol{\mu}_k\}$ to some values
- **Step 1** Assume the current value of $\{\boldsymbol{\mu}_k\}$ fixed, minimize J over $\{r_{nk}\}$, which leads to the following cluster assignment rule

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

Lloyd's Algorithm for k-means Clustering

Minimize distortion measure alternative optimization between $\{r_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$

- **Step 0** Initialize $\{\boldsymbol{\mu}_k\}$ to some values
- **Step 1** Assume the current value of $\{\boldsymbol{\mu}_k\}$ fixed, minimize J over $\{r_{nk}\}$, which leads to the following cluster assignment rule

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

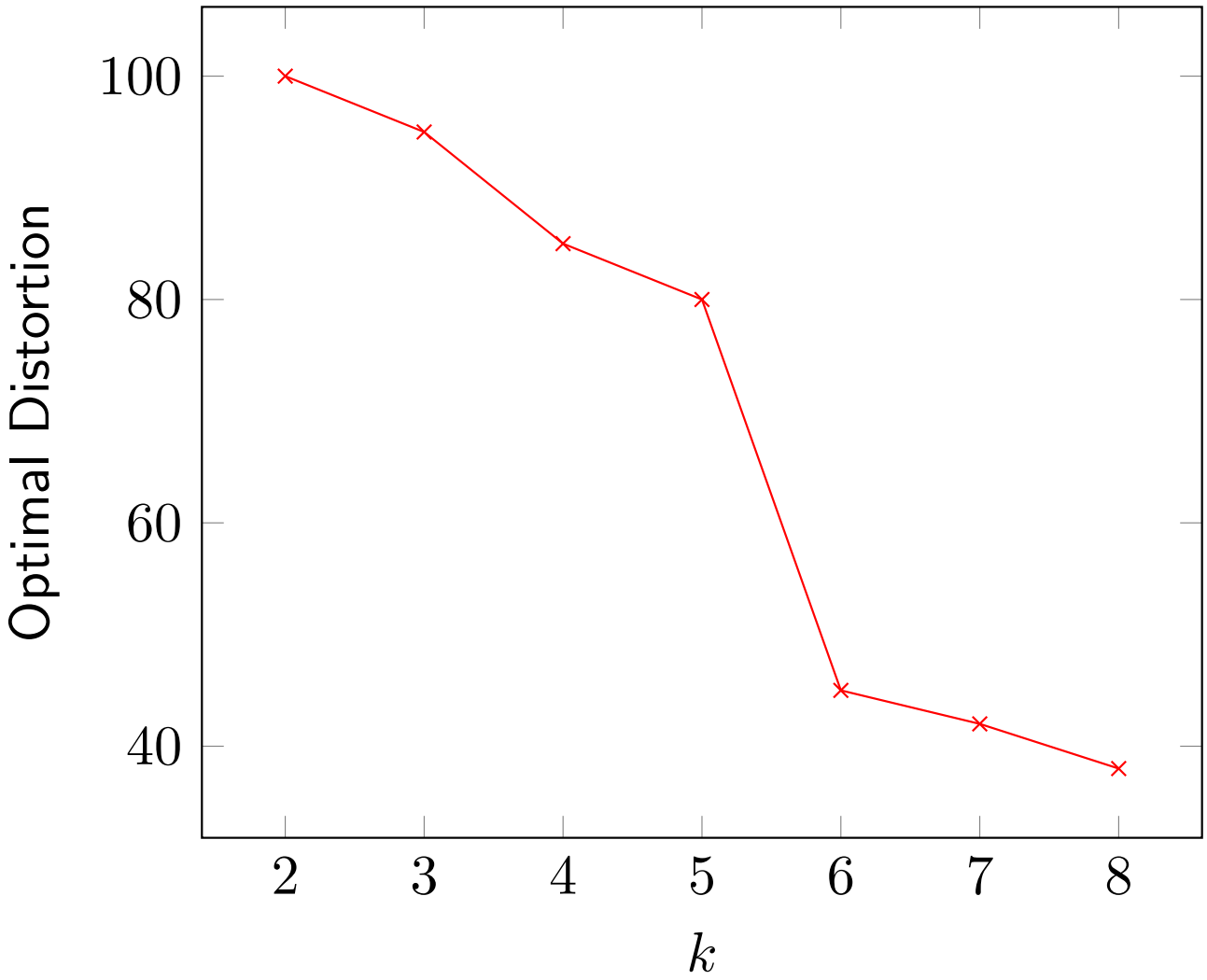
- **Step 2** Assume the current value of $\{r_{nk}\}$ fixed, minimize J over $\{\boldsymbol{\mu}_k\}$, which leads to the following rule to update the prototypes of the clusters

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\boldsymbol{x}_n}{\sum_n r_{nk}}$$

- **Step 3** Determine whether to stop or return to Step 1

Outline

How should we choose k?



What do you suppose the "natural" cluster count for this data is?

But Lloyd's algorithm is inaccurate sometimes!

- Observation: Lloyd's tends to be good on natural problems
- For determining k, we aren't computing optimal distortion
- Does this help or hurt our case for estimating k?

Acknowledgements

- Section "Finding good solutions to clustering" is based on
  "The Effectiveness of Lloyd-Type Methods for the k-Means Problem"
  by Ostrovsky, Rabani, Schulman, Swamy
  Appeared in FOCS 2006

- Section "Clustering Big Data" is based on
  "Streaming k-means approximation" by Ailon, Jaiswal, Monteleoni
  Appeared in NIPS 2009

- Papers can be found on Google Scholar if desired;
- This presentation is not intended as an in-depth study of these papers.

Outline

Lloyd's disconnect: Theory vs Practice

- Lloyd's Algorithm for k-means :
  - ... works very well in practice.
  - ... can be arbitrarily bad in cost
  - ... can take exponential time to converge
- So it's bad in theory, good in practice.
- What should we do?

What is a meaningful clustering anyway?

- In general, we don't view k-means as geometry problem.
- What if two very different ways to partition are near $\Delta_k^2(X)$

- What if $\Delta_k^2(X) \approx \Delta_{k-1}^2(X)$?

- *Should we care* if k-means difficult in this circumstance?

Separation Condition

Define that data set X is ϵ-separated for k-means if:

$$\Delta_k^2(X)/\Delta_{k-1}^2(X) \leq \epsilon^2$$

Why does this make sense?

Algorithm for 2-means

- Select starting position.
  - Lloyd's: select any two points.
  - Instead: pick pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Call the chosen points $\hat{c}_1$ and $\hat{c}_2$
  - This biases distribution towards pairs that contribute a lot to $\Delta_1^2(X)$
  - This is likely to select from the *cores* of the two optimal clusters.

Algorithm for 2-means

- Select starting position.
  - Lloyd's: select any two points.
  - Instead: pick pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Call the chosen points $\hat{c}_1$ and $\hat{c}_2$
  - This biases distribution towards pairs that contribute a lot to $\Delta_1^2(X)$
  - This is likely to select from the *cores* of the two optimal clusters.
- Recenter the clusters
  - Lloyd's: centers of mass
  - Instead: Instead of using full partition, use only those points within radius $\|\hat{c}_1 - \hat{c}_2\|/3$ of $\hat{c}_i$.

Algorithm for 2-means

- Select starting position.
  - Lloyd's: select any two points.
  - Instead: pick pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Call the chosen points $\hat{c}_1$ and $\hat{c}_2$
  - This biases distribution towards pairs that contribute a lot to $\Delta_1^2(X)$
  - This is likely to select from the *cores* of the two optimal clusters.
- Recenter the clusters
  - Lloyd's: centers of mass
  - Instead: Instead of using full partition, use only those points within radius $\|\hat{c}_1 - \hat{c}_2\|/3$ of $\hat{c}_i$.
- Repeat?
  - Lloyd's: either set # of times, or until stable.
  - Instead: this is enough.

- Let r_i^2 be the error from cluster i.
- **Claim**: $\max(r_1^2, r_2^2) \leq \frac{\epsilon^2}{1-\epsilon^2}\|c_1 - c_2\|^2$
  (where c_i are the optimal centers)

$$\Delta_1^2(X) = \Delta_2^2(X) + \frac{n_1 n_2}{n} \cdot \|c_1 - c_2\|^2$$

$$\frac{n_1 n_2}{n} \cdot \Delta_2^2(X) = \|c_1 - c_2\|^2 \frac{\Delta_2^2(X)}{\Delta_1^2(X) - \Delta_2^2(X)}$$

$$r_1^2 \cdot \frac{n}{n_2} + r_2^2 \cdot \frac{n}{n_2} \leq \frac{\epsilon^2}{1-\epsilon^2}\|c_1 - c_2\|^2$$

Remember separation condition

Separation $\Delta_k^2(X) \leq \epsilon^2 \Delta_{k-1}^2(X)$

- Stage one: seeding.
  - At the end of this stage, k initial in cores
- Stage two: re-centering

Sampling-based seeding

- First method: sampling
  - Pick first pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Then $\hat{c}_{i+1} \in X$ with probability $\min_{j \leq i} \|x - \hat{c}_j\|^2$
  - Easy to implement

Sampling-based seeding

- First method: sampling
  - Pick first pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Then $\hat{c}_{i+1} \in X$ with probability $\min_{j \leq i} \|x - \hat{c}_j\|^2$
  - Easy to implement
- Second method: greedy delete
  - Start with all points are centers
  - Delete until only k remain.

Sampling-based seeding

- First method: sampling
  - Pick first pair $x, y \in X$ with probability proportional to $\|x - y\|^2$
  - Then $\hat{c}_{i+1} \in X$ with probability $\min_{j \leq i} \|x - \hat{c}_j\|^2$
  - Easy to implement
- Second method: greedy delete
  - Start with all points are centers
  - Delete until only k remain.
- Third method: a combination
  - Perform second method, but instead of all points, pick $\frac{2k}{1-5\sqrt{\epsilon}} + \frac{2\ln(2/\sqrt{\epsilon})}{(1-5\sqrt{\epsilon})^2}$ random points.
  - Choose this many points using first method.
  - Weigh each chosen point by its cluster size from X
  - Now run second method.

Re-centering

- We have a clustering.
- Lloyd's will re-position to center of mass.
- Two methods here too.
- Method one: center of mass, but use only "nearby" points
- Method two: sampling-based

Conclusion

- If data is "well-clusterable"
  Then Lloyd-style methods perform well for a reason
  and are provably near-optimal.
- Given separation condition, we can get good clustering:
  - Hybrid seeding mechanism
  - Core-based re-positioning
- The exact guarantee, not derived in lecture
  - Cost found is at most $\frac{1-\epsilon^2}{1-37\epsilon^2} \cdot \Delta_k^2(X)$
  - With probability $1 - O(\sqrt{\epsilon})$
  - Computed in total time $O(nkd + k^3 d)$

Outline

Big data issue

- Many algorithms from class don't scale well.
- The k-means algorithms read the data repeatedly
- This is the so-called "batch setting."

Another way to get a low-cost clustering

Authors call this "k-means#

1. Choose $3 \log k$ centers independently and uniformly at random.
2. Repeat $k - 1$ times the following:
    1. Choose $3 \cdot \log k$ centers independently with probability as per earlier

- This provides a set of centers within $O(1) \cdot \Delta_k^2(X)$
- Of course, it isn't precisely a fair comparison

Single-pass algorithm

- Separate X into sets of size $\sqrt{nk}$
- For each set, create $3k \log k$ centers with k-means#
- Store each output as weighted set of points.
- Combine the sets with some reasonable k-means algorithm

Improved memory-approximation tradeoffs

- The previous algorithm uses some amount of memory
- Every "level" of k-means usage costs us in approximation guarantee
- There is some trade-off in memory requirements vs guarantee