Georgia Tech

DATA ANALYTICS USING DEEP LEARNING

GT 8803 // FALL 2018 // JACOB LOGAS

LECTURE #10: LOCALITY-SENSITIVE HASHING FOR EARTHQUAKE DETECTION

CREATING THE NEXT®

TODAY'S PAPER

- Locality-Sensitive Hashing for Earthquake Detection: A Case Study of Scaling Data-Driven Science
  - End-to-end earthquake detection pipeline
  - Fingerprinting for compact representation
  - Domain knowledge for optimization
  - Concise detection results

TODAY'S PAPER

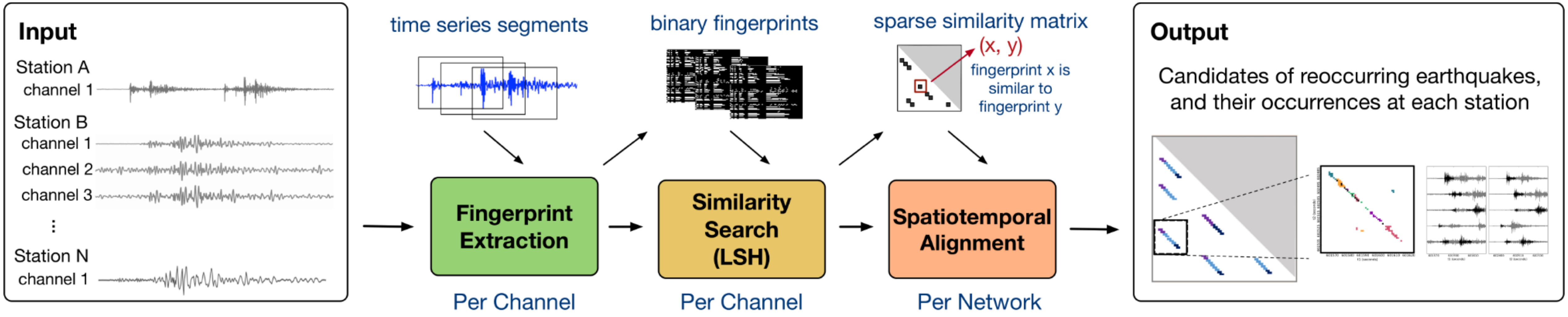


Figure 2: The three steps of the end-to-end earthquake detection pipeline: fingerprinting transforms time series into binary vectors (Section 5); similarity search identifies pairs of similar binary vectors (Section 6); alignment aggregates and reduces false positives in results (Section 7).

Figure from [1]

TODAY'S AGENDA

- Motivation
- Background
- Problem Overview
- Key Idea
- Technical Details
- Experiments
- Discussion

MOTIVATION

- Large amount of earthquake data
  - High frequency sensor data
  - Multiple sensor sites
- Small fraction of earthquakes cataloged
  - Traditionally done manually
- Difficult to detect at low magnitudes
  - True earthquakes get lost in noise
  - Uncover unknown seismic sources

PREVIOUS WORK

- Audio Fingerprinting
  - Links short, unlabeled, snippets of audio to data
  - Process audio as image
- Fingerprint And Similarity Thresholding (FAST)
  - Based on waveform similarity
  - Applies Locality Sensitive Hashing (LSH)
  - Difficult to scale beyond 3 months of data
  - Runtime is near quadratic with input size
  - Seismologists still cannot make use of all data

NAIVE SEARCH

- Waveform Similarity
  - Use template waveforms from catalogs
  - Measure similarity using cross-correlation
- Brute-Force Blind
  - Doesn’t require templates
  - Searches for similar waveform sets
  - Quadratic

WAVEPRINT

- Audio fingerprinting for compact representation
- LSH and Hamming distance for retrieval
- Unsupervised
- Method:
  1. Convert audio to spectrogram
  2. Create spectral images
  3. Extract top Haar-wavelets according to magnitude
  4. Wavelet signature computed
  5. Select top t wavelets (by magnitude)

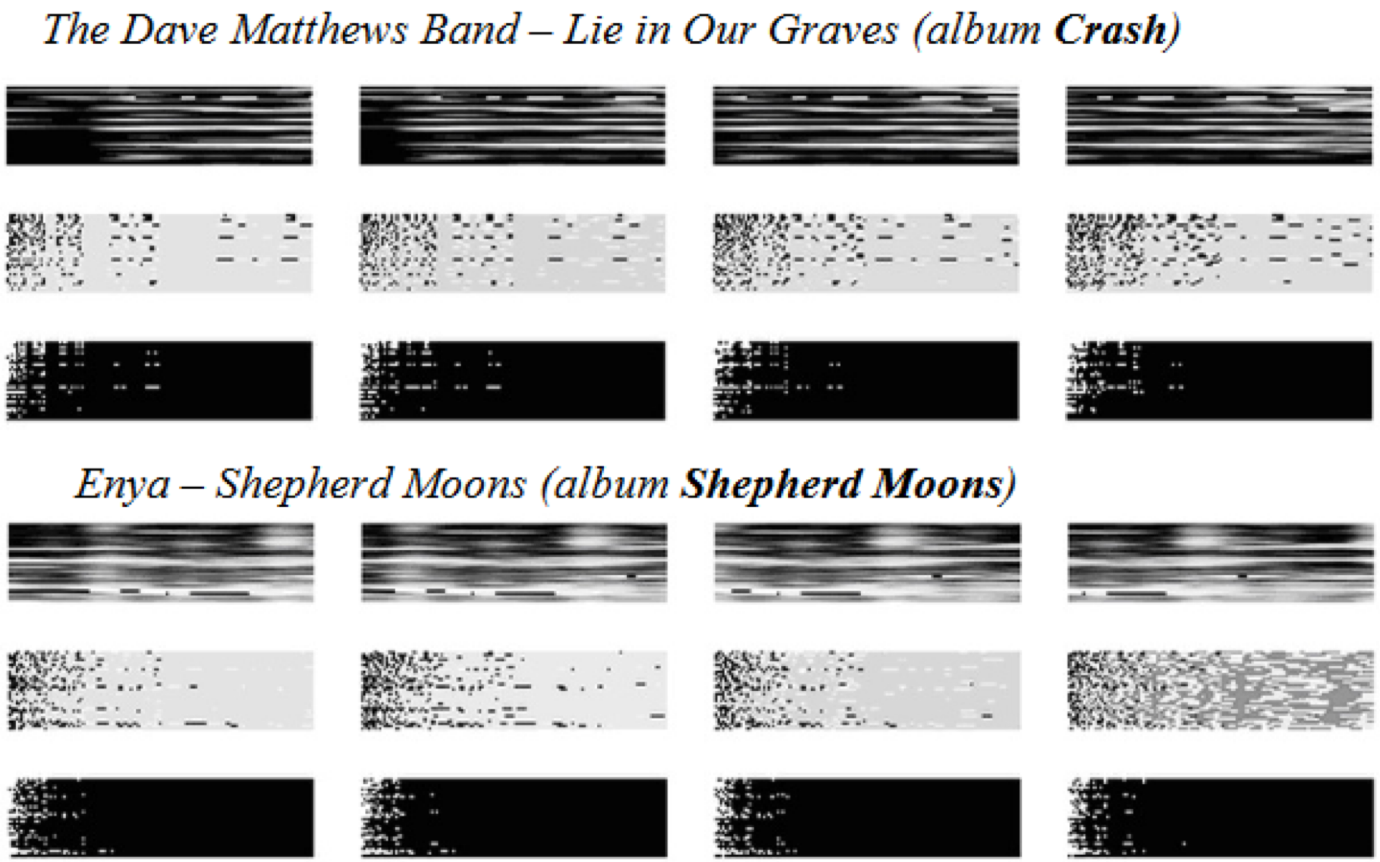
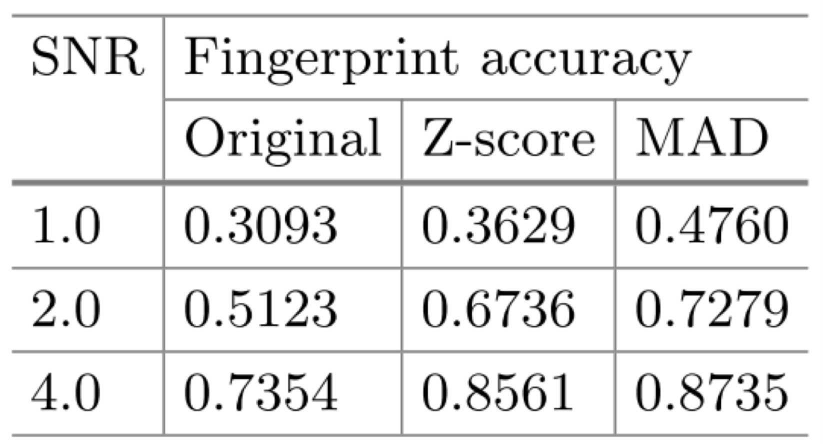


Figure 1. The representation for two songs – 4 consecutive spectrogram images shown for each, skipping 200 ms. For each song, top row: original spectrogram image, second row: wavelet magnitudes; third row: the top-200 wavelets. Note that the top wavelets have a distinctive pattern for each of the songs.

Figure from [4]

FAST

- Detect event by identifying similar waveforms
- Modeled after aforementioned system
  - Create fingerprint from waveform
  - Perform approximate similarity search with LSH



SNR	Fingerprint accuracy		
	Original	Z-score	MAD
1.0	0.3093	0.3629	0.4760
2.0	0.5123	0.6736	0.7279
4.0	0.7354	0.8561	0.8735

Median Jaccard similarity of clean and low-SNR earthquake waveforms

Figure from [3]

FAST

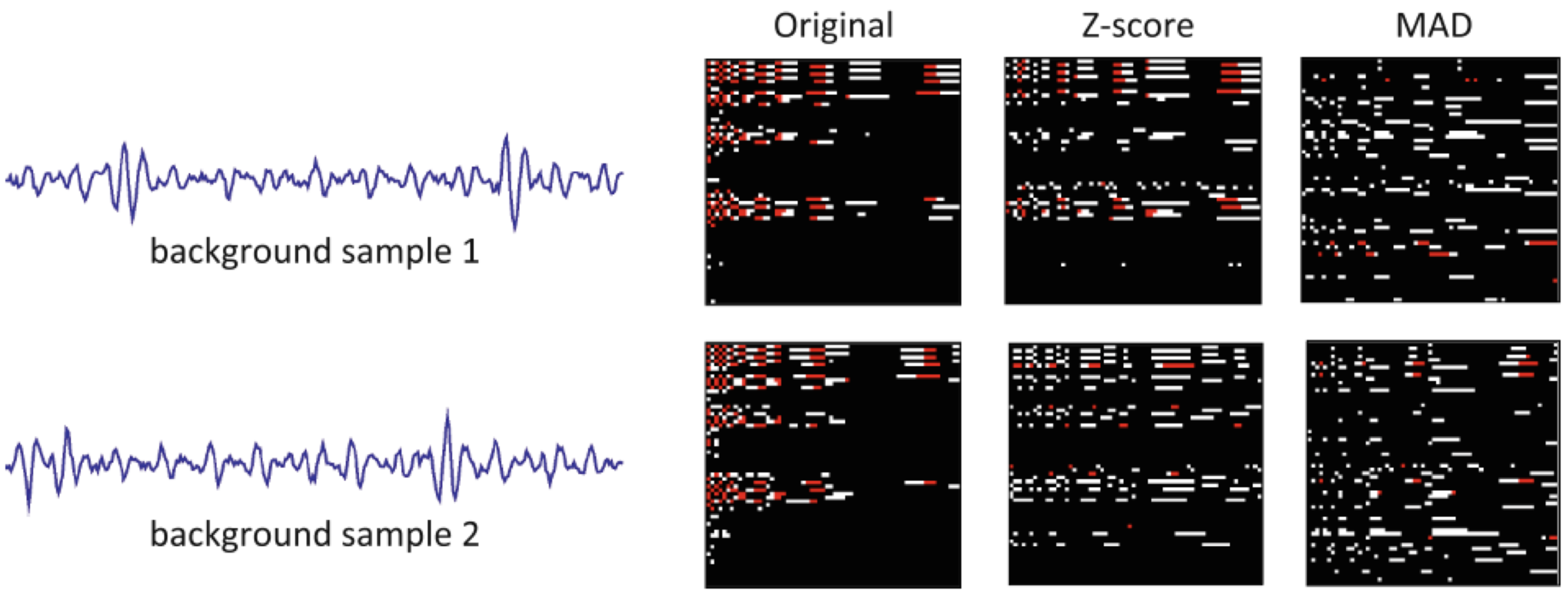


Fig. 3. Comparison of fingerprinting schemes applied to background noise. The Jaccard similarities between the fingerprints are: 0.266 (original), 0.117 (Z-score), and 0.040 (MAD).

Figure from [3]

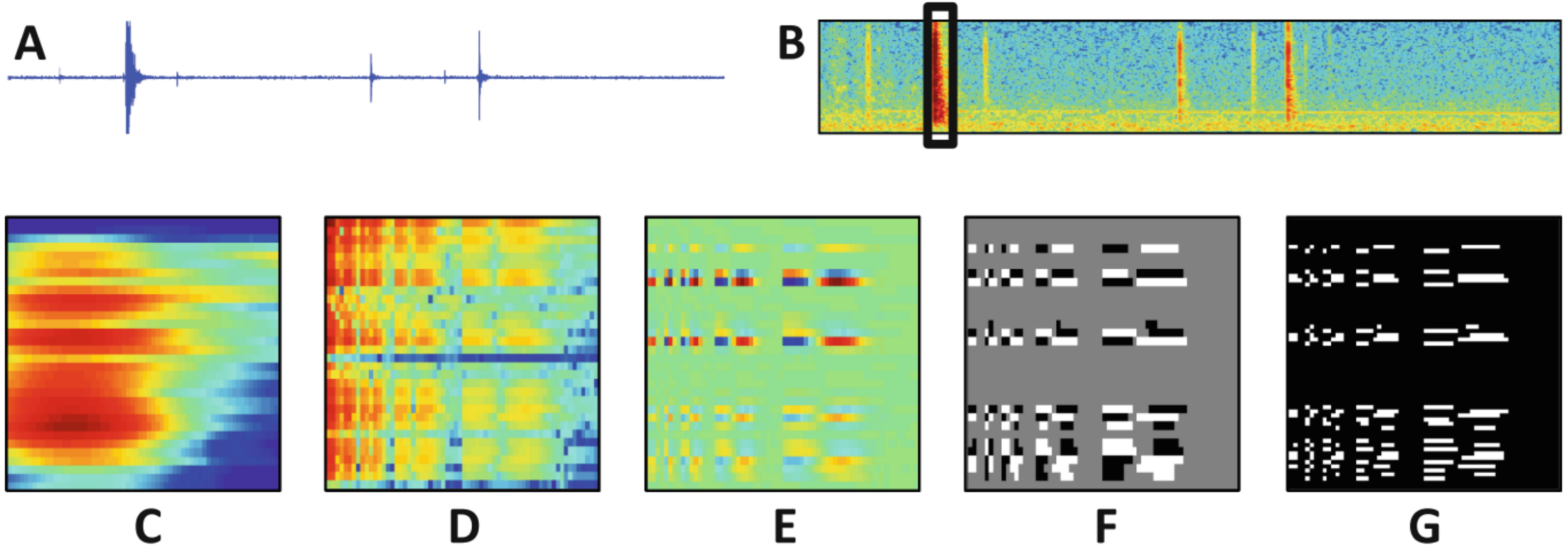


Fig. 2. Feature Extraction process in FAST: (A) continuous data, (B) spectrogram, (C) spectral image, (D) discrete Haar wavelet transform, (E) adjusted wavelet coefficients, (F) coefficient selection, (G) conversion to binary fingerprint

Figure from [3]

LOCALITY-SENSITIVE HASHING

- Near neighbor search
- High dimensional space
- Partition space according to some heuristic
- Try to hash near neighbors in same buckets
- $O(n^{\frac{1}{c}})$ for c approximation
- Naïve uses $O(n * d)$ where d is dimension

Slides on this LSH algorithm from a talk given by Piotr Indyk

LSH SIMILARITY SEARCH

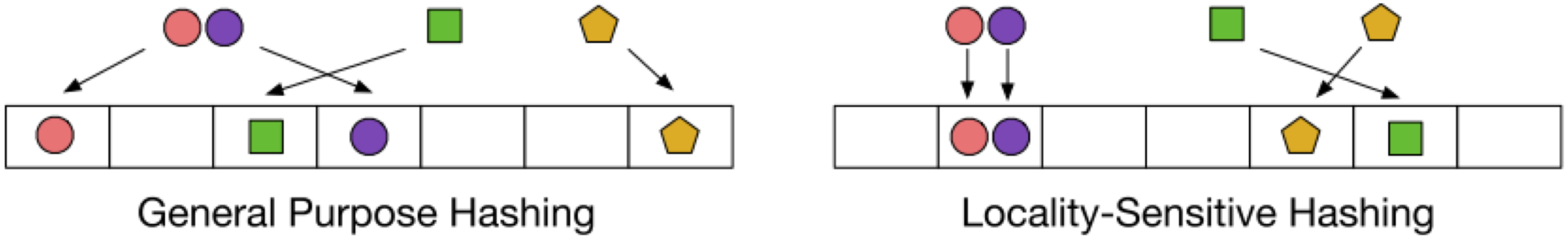


Figure 4: Locality-sensitive hashing hashes similar items to the same hash “bucket" with high probability.

Figure from [1]

PROBLEM OVERVIEW

- Decades of earthquake data
- FAST doesn’t scale beyond 3 months
- Actual LSH runtime grows near quadratic
  - Due to correlations in seismic signals
- 5x dataset causes 30x greater query time
- Similar, non-earthquake, noise is falsely matched
  - Adds to overall search complexity

KEY IDEAS

- Improve FAST efficiency using
  - Systems
  - Algorithms
  - Domain expertise
- End-to-end detection pipeline
  1. Fingerprint extraction
  2. Apply LSH on binary fingerprints
  3. Alignment to reduce result size improving readability

FINGERPRINT EXTRACTION

- Basically the same as previously discussed
- Follows 5 steps:
  1. Spectrogram
  2. Wavelet Transform
  3. Normalization
  4. Top coefficient
  5. Binarize
- An important optimization made

FINGERPRINT EXTRACTION

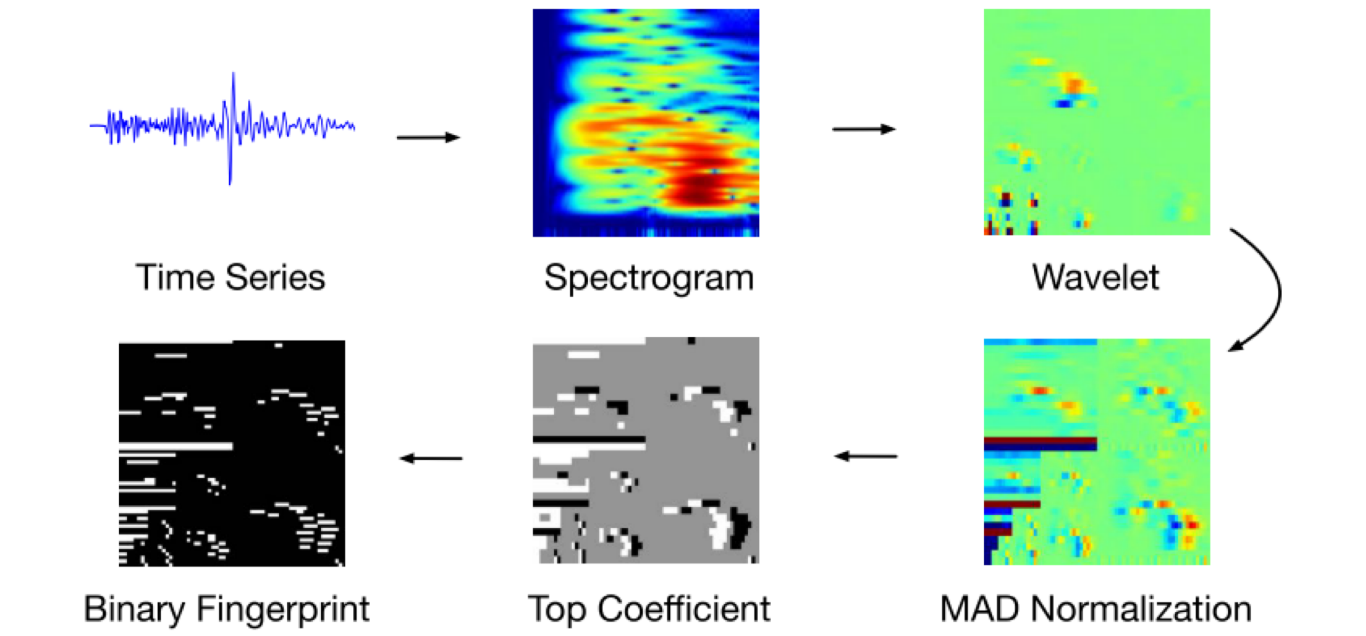


Figure 3: The fingerprinting algorithm encodes time-frequency features of the original time series into compact binary vectors.

Figure from [1]

OPT: MAD VIA SAMPLING

- Fingerprinting is linear in complexity
  - Years of data takes several days on single core
- Normalization takes two passes over data
  1. Get median and MAD
  2. Normalize fingerprint wavelets (parallelizable)
- First pass is the bottleneck here
  - To alleviate, approximate true median and MAD
  - MAD confidence interval shrinks with $n^{\frac{1}{2}}$
  - Sampling 1% or less of input for long durations suffices

LSH SIMILARITY SEARCH

- MinHash LSH on binary fingerprints
  - Random projection from high to lower dim
  - Hash similar items to same bucket with high Pr
  - Compares only to fingerprints sharing bucket
- Limits
  - Signature generation: poor memory locality
  - MinHash: only keeps min value for each map
  - High Collisions: elements aren't independent
  - Large Hash Table: exceed main memory
  - Noise as earthquakes: false positives due to noise similar to earthquakes

OPT: MODIFYING GEN LOOP

- MinHash
  - First non-zero of fingerprint under random permutation
  - Permutation: mapping elements to random indices
  - Sparse input induces cache misses
- Block access to hash mappings
  - Use fingerprint dimensions in place of hash function
  - Lookups for non-zero elements blocked in rows

OPT: USE MIN-MAX HASH

- Keeps both min and max for each mapping
- Reduces required hash functions by ½
- Unbiased estimator of similarity
- Can achieve similar/smaller MSE in practice

OPT: ALLEVIATE COLLISIONS

- Poor distribution of hash signatures
  - Large buckets or high selectivity
  - All fingerprints in same bucket, search is $O(n^2)$
- Fingerprints not necessarily independent
  - LSH working as advertised (maybe a little too well)
- LSH hyperparameters tuned
  - Increasing hash function number reduces collision
  - Reduce false matches by scaling up hash table number

FINGERPRINT PR

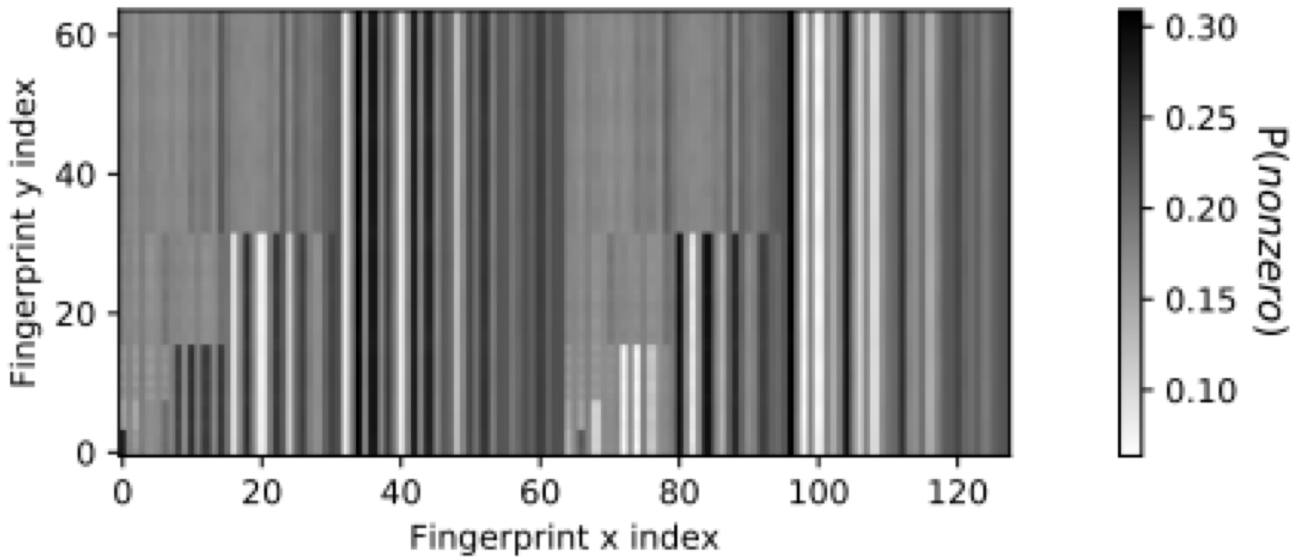


Figure 5: Probability that each element in the fingerprint is equal to 1, averaged over 15.7M fingerprints, each of dimension 8192, generated from a year of time series data. The heatmap shows that some elements of the fingerprint are much more likely to be non-zero compared to others.

Figure from [1]

OPT: PARTITIONING

- Total size of hash signatures ~250GB
- To scale, perform similarity search in partitions
  - Evenly partition fingerprints
- Populate hash tables one partition at a time
  - Keep lookup table in memory
- During query, output matches over all other fingerprints for only current partition
  - Same output with only subset of fingerprints in mem
- Allows for parallelization of hash signature gen and querying

OPT: DOMAIN-SPECIFIC FILTERS

- Stations can have repeating narrow-band noise
  - Can be falsely identified as earthquake candidates
- Filtering irrelevant frequencies
  - Bandpass filter for bands with high amplitudes containing low seismic activities
  - Selected manually through examination
  - Cutoff spectrograms at corner of bandpass filter
- Remove correlated noise
  - Repetitive noise occurs in bands with earthquake signals
  - Give NN matches dominating similarity search
  - If many NN matches in short time, filter out

SPATIOTEMPORAL ALIGNMENT

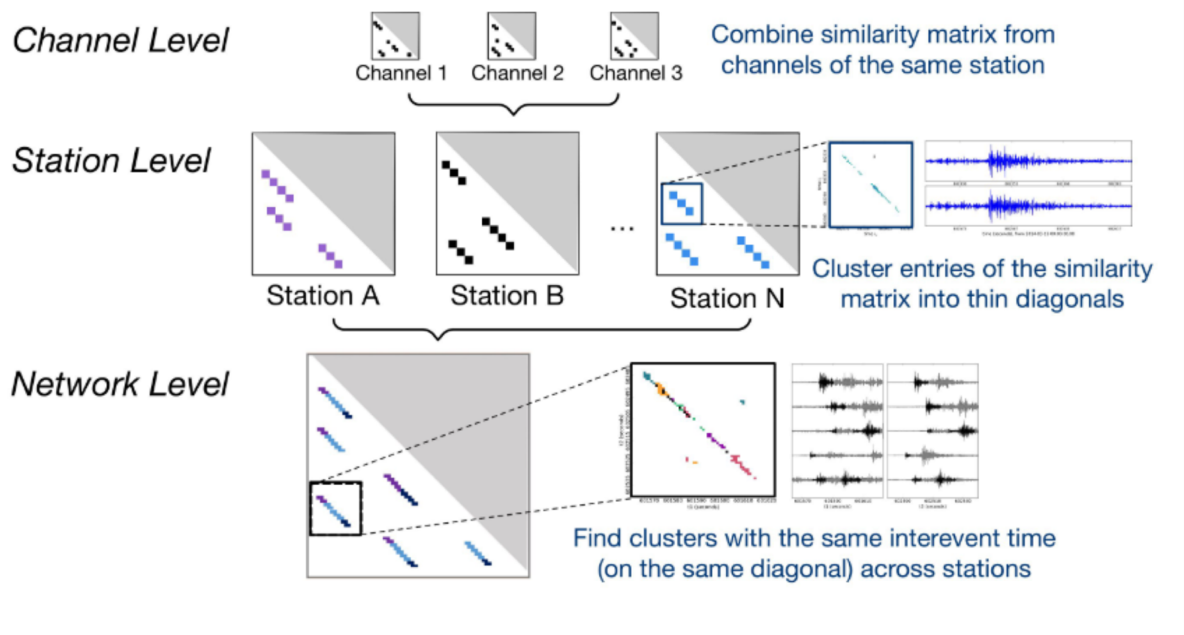

Channel Level
Channel 1
Channel 2
Channel 3
Combine similarity matrix from channels of the same station
Station Level
Station A
Station B
...
Station N
Cluster entries of the similarity matrix into thin diagonals
Network Level
Find clusters with the same interevent time (on the same diagonal) across stations

Figure 8: The alignment procedure combines similarity search outputs from all channels in the same station (Channel Level), groups similar fingerprint matches generated from the same pair of reoccurring earthquakes (Station Level), and checks across seismic stations to reduce false positives in the final detection list (Network Level).

Figure from [1]

SPATIOTEMPORAL ALIGNMENT

- Search outputs pairs from input
  - Doesn’t determine if pairs actual earthquakes
  - One year can generate more than 5 million pairs
- Domain knowledge used to reduce output size
- Output is optimized at different levels
  - Channel
  - Station
  - Network

CHANNEL LEVEL

- Channels at same station experience movement at same time
- Merge channel detection events at each station
  - Fingerprint matches tend to occur across channels
  - Noise may only exist in some channels
  - This adds a higher similarity threshold
  - Prunes false positives while maintaining weak matches

STATION LEVEL

- Similarity matrix diagonals represent earthquakes
  - Corresponds to group of similar fingerprint pairs
  - Separated by a constant offset (inter-event time)
- Exclude self-matches generated from overlapping
- After grouping diagonals
  - Reduce cluster to summary statistics
- Significantly reduce output size

NETWORK LEVEL

- Earthquakes visible across network of sensors
  - Travel time only function of distance, not magnitude
  - Thus fixed travel time between network nodes
- Diagonals with station Δt are same event
- Earthquake must be seen n times for detection
- Postprocessing reduce from ~2Tb of pairs to 30K timestamps

END-TO-END

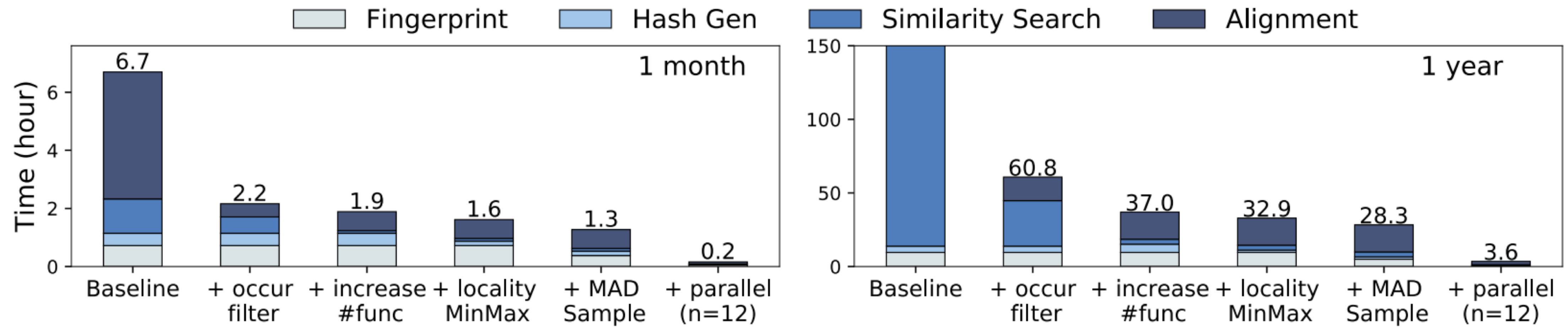


Figure 10: Factor analysis of processing 1 month (left) and 1 year (right) of 100Hz data from LTZ station in the New Zealand dataset. We show that each of our optimization contributes to the performance improvements, and enabled an over 100× speed up end-to-end.

Figure from [1]

LSH RUNTIME

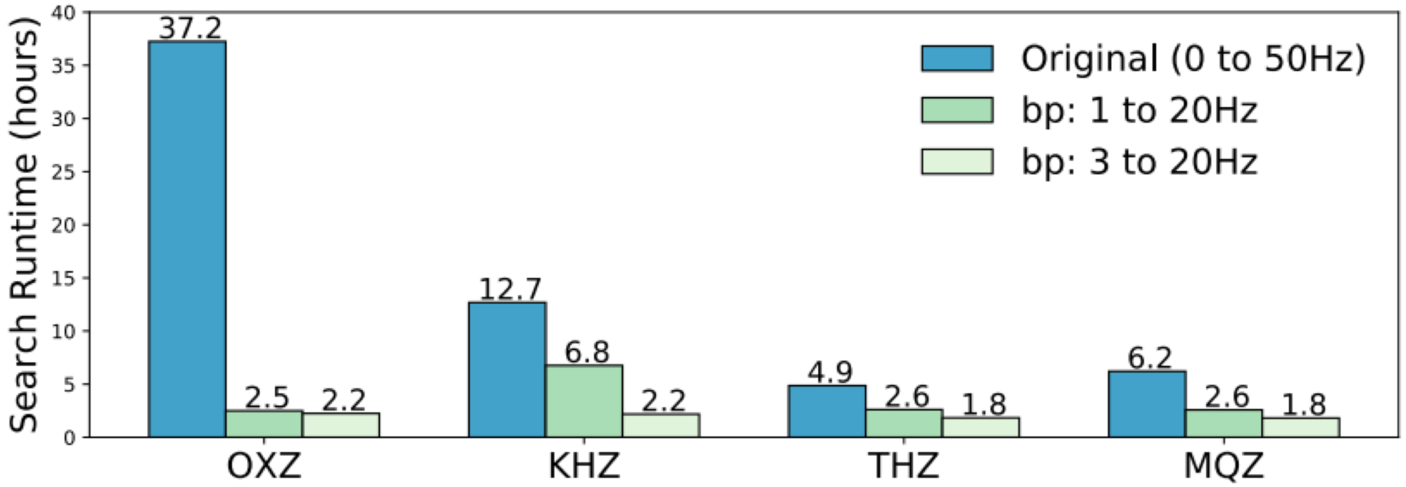


Figure 11: LSH runtime under different band pass filters. Matches of noise in the non-seismic frequency bands can lead to a 16× increase in runtime and over 200 × increase in output size for unfiltered time series.

Figure from [1]

LSH RUNTIME

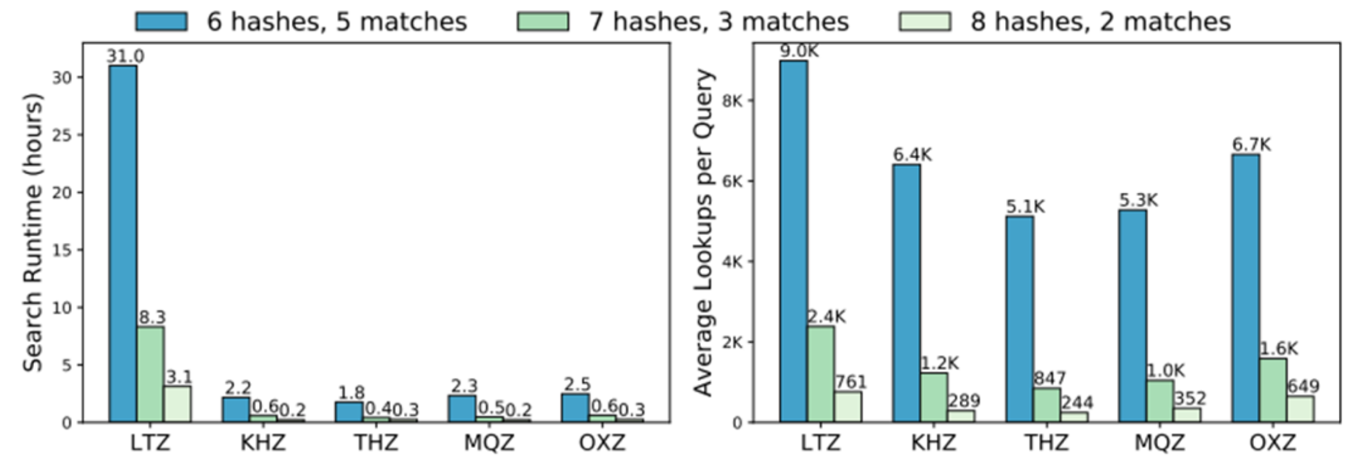


Figure 12: Effect of LSH parameters on similarity search runtime and average query lookups. Increasing the number of hash functions significantly decreases average number of lookups per query, which results in an up to 10× improvement in runtime.

Figure from [1]

LSH PARTITIONING

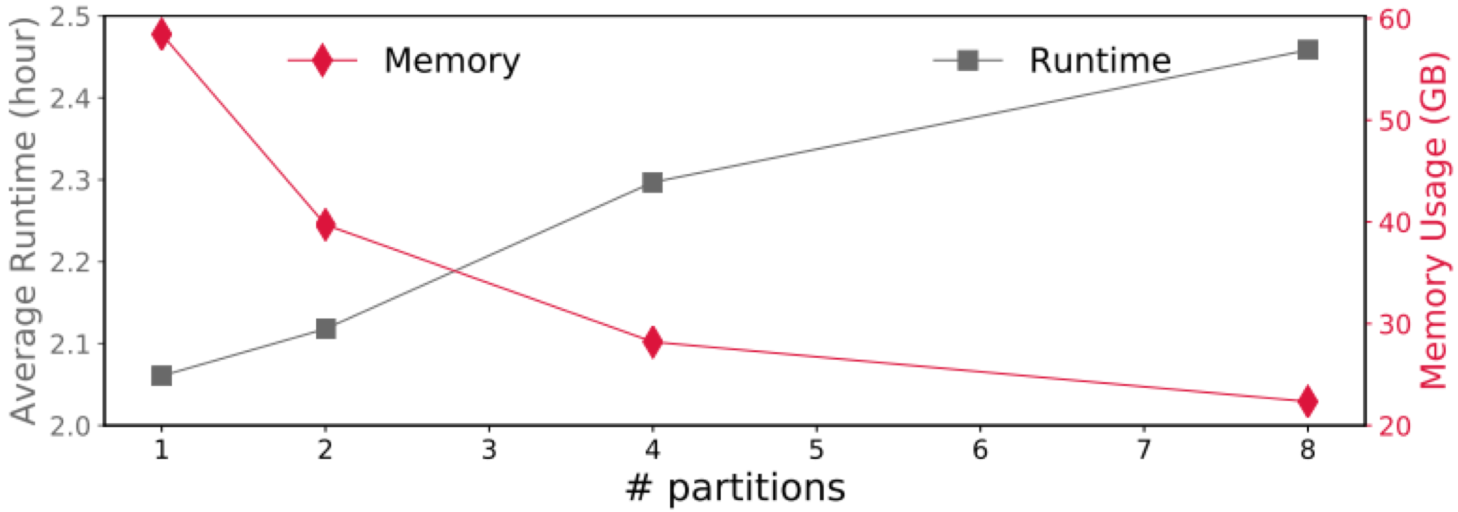
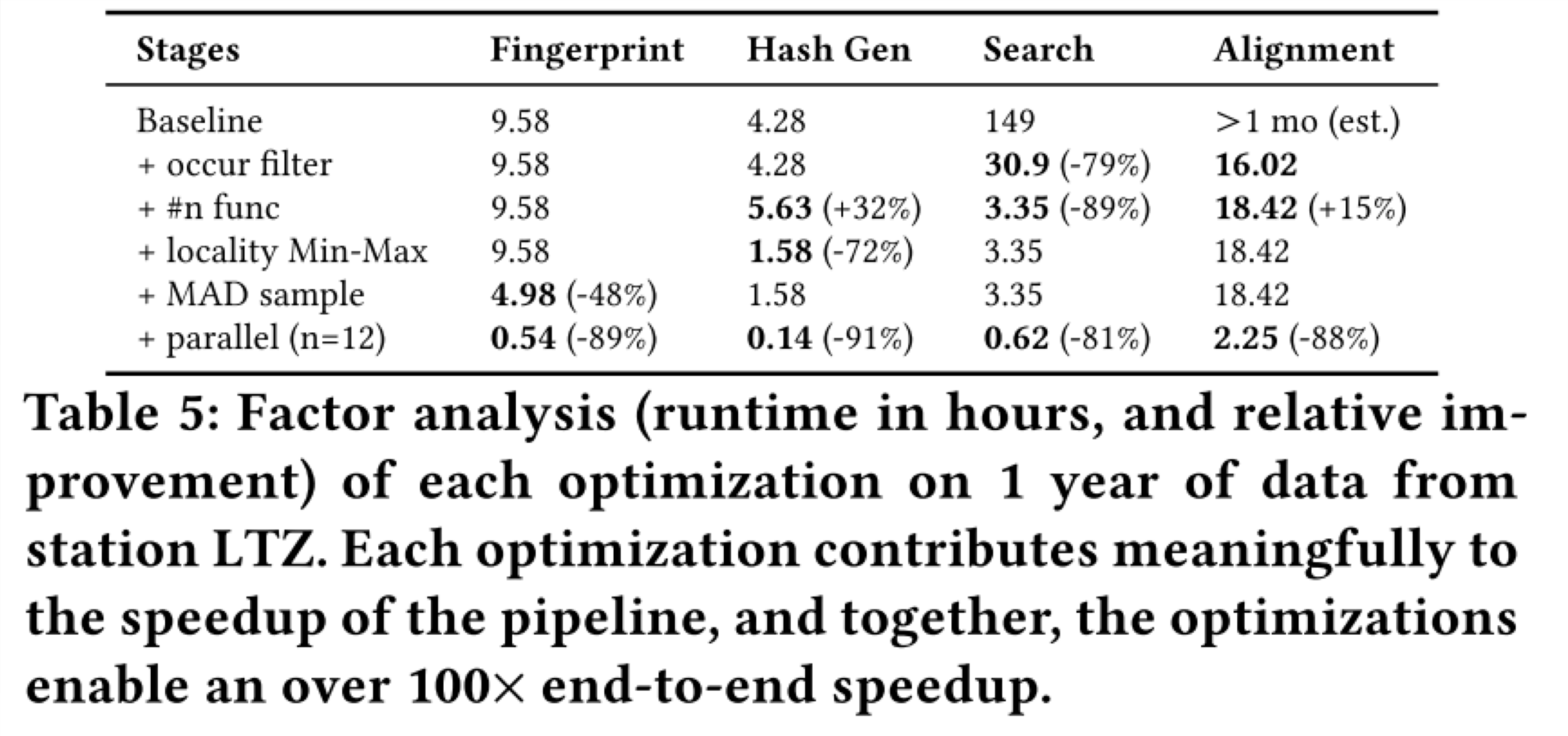


Figure 13: Runtime and memory usage for similarity search under a varying number of partitions. By increasing the number of search partitions, we are able to decrease the memory usage by over 60% while incurring less than 20% runtime overhead.

Figure from [1]

OVERALL SYSTEM SPEEDUP



Stages	Fingerprint	Hash Gen	Search	Alignment
Baseline	9.58	4.28	149	>1 mo (est.)
+ occur filter	9.58	4.28	**30.9** (-79%)	**16.02**
+ #n func	9.58	**5.63** (+32%)	**3.35** (-89%)	**18.42** (+15%)
+ locality Min-Max	9.58	**1.58** (-72%)	3.35	18.42
+ MAD sample	**4.98** (-48%)	1.58	3.35	18.42
+ parallel (n=12)	**0.54** (-89%)	**0.14** (-91%)	**0.62** (-81%)	**2.25** (-88%)

Table 5: Factor analysis (runtime in hours, and relative improvement) of each optimization on 1 year of data from station LTZ. Each optimization contributes meaningfully to the speedup of the pipeline, and together, the optimizations enable an over 100× end-to-end speedup.

IMPACT OF SYSTEM

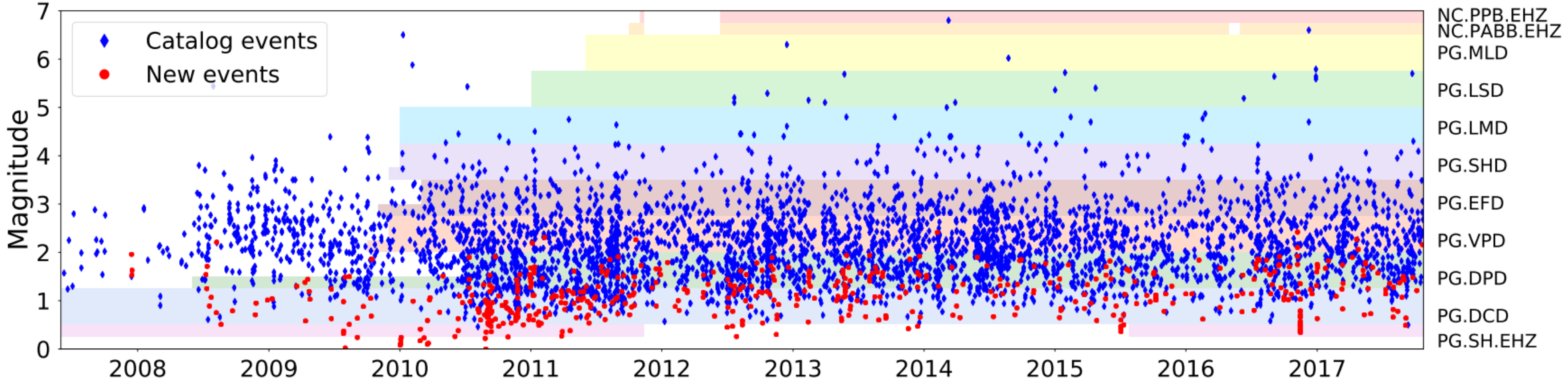


Figure 15: The left axis shows origin times and magnitude of detected earthquakes, with the catalog events marked in blue and new events marked in red. The colored bands in the right axis represent the duration of data used for detection collected from 11 seismic stations and 27 total channels. Overall, we detected 3957 catalog earthquakes (diamond) as well as 597 new local earthquakes (circle) from this dataset.

Figure from [1]

STRENGTHS

- Using domain knowledge for optimization
- Pipeline able to detect difficult earthquakes
- Good speedup allowing for use of entire dataset
- Filter out many noisy signals

WEAKNESSES

- Not directly generalizable to other domains
- LSH strained, needed many optimizations
- Not developed for distributed systems
- Not all optimizations implemented
- Little validation information

DISCUSSION

- LSH Alternatives
- Insights
- Applications
- Generalizability

REFERENCES

1. Kexin Rong, Clara E. Yoon, Karianne J. Bergen, Hashem Elezabi, Peter Bailis, Philip Levis, and Gregory C. Beroza. 2018. Locality-Sensitive Hashing for Earthquake Detection: A Case Study Scaling Data-Driven Science. https://doi.org/arXiv:1803.09835v2
2. Wei Dong, Zhe Wang, William Josephson, Moses Charikar, and Kai Li. 2008. Modeling LSH for performance tuning. In *Proceeding of the 17th ACM conference on Information and knowledge mining - CIKM '08*, 669. https://doi.org/10.1145/1458082.1458172
3. Karianne Bergen, Clara Yoon, and Gregory C. Beroza. 2016. Scalable Similarity Search in Seismology: A New Approach to Large-Scale Earthquake Detection. . Springer, Cham, 301–308. https://doi.org/10.1007/978-3-319-46759-7_23
4. Shumeet Baluja and Michele Covell. 2007. Audio fingerprinting: Combining computer vision & data stream processing. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, II-213-II-216. https://doi.org/10.1109/ICASSP.2007.366210