2023

SPRING IO 10TH ANNIVERSARY

#springio23

Vector Similarity Search

in Spring with Redis Stack

Brian Sam-Bodden

redis

BARCELONA MAY 18-19 / WWW.SPRINGIO.NET

bsb@redis.com

@bsbodden

github.com/bsbodden

Vector Similarity Search

TLDR

Why? Vast majority of data is Unstructured data!

What? Vector Databases store vectors efficiently

How? Redis provides Vector Storage and Search Capabilities

Vector Similarity Search

TLDR

Why? Vast majority of data is Unstructured data!

What? Vector Databases store vectors efficiently

How? Redis provides Vector Storage and Search Capabilities

Vector Similarity Search

TLDR

Why? Vast majority of data is Unstructured data!

What? Vector Databases store vectors efficiently

How? Redis provides Vector Storage and Search Capabilities

Vector Similarity Search

TLDR

Why? Vast majority of data is Unstructured data!

What? Vector Databases store vectors efficiently

How? Redis provides Vector Storage and Search Capabilities

REmote DIctionary Server

REmote DIctionary Server

REmote DIctionary Server

REmote DIctionary Server

In-memory First

Optionally Persistent

In-memory First

Optionally Persistent


OSS


OSS

1

1

Strings

1

Sets

Strings

1

Sets

Strings

Lists

1

Sets

Hashes

Strings

Lists

1

Sets

Hashes

Strings

Lists

Bitmaps

1

Sets	Hashes	Sorted Sets
Strings	Lists	Bitmaps

1

Sets | Hashes | Sorted Sets

Strings | Lists | Bitmaps | Geospatial

1

Sets	Hashes	Sorted Sets	Bit Field
Strings	Lists	Bitmaps	Geospatial

1

Sets | Hashes | Sorted Sets | Bit Field

Strings | Lists | Bitmaps | Geospatial | Streams

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

Foundational Data Structures

OSS

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

Foundational Data Structures

1

Sets | Hashes | Sorted Sets | Bit Field | HyperLogLog

Strings | Lists | Bitmaps | Geospatial | Streams

Foundational Data Structures

2

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

Foundational Data Structures

2 JSON

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

Foundational Data Structures

2

JSON

Probabilistic

1

Sets	Hashes	Sorted Sets	Bit Field	HyperLogLog
Strings	Lists	Bitmaps	Geospatial	Streams

Foundational Data Structures

2

JSON

Probabilistic

Documents and Probabilistic Data Structures

1

Sets | Hashes | Sorted Sets | Bit Field | HyperLogLog

Strings | Lists | Bitmaps | Geospatial | Streams

Foundational Data Structures

3

2

JSON

Probabilistic

Documents and Probabilistic Data Structures

1

Sets | Hashes | Sorted Sets | Bit Field | HyperLogLog

Strings | Lists | Bitmaps | Geospatial | Streams

Foundational Data Structures

3 Search

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

3 Search Functions

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

3 Search Functions

Redis-side Computing

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

4

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

OSS

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

4 redisinsight

STACK

3 Search Functions

Redis-side Computing

2 JSON Probabilistic

Documents and Probabilistic Data Structures

OSS

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

4 redisinsight redisōM

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

OSS

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

4 redisinsight redisOM Java Jedis

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

OSS

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

4 redisinsight redis OM Java Jedis node-redis

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

OSS

Foundational Data Structures

4 redisinsight redis OM Java Jedis node-redis redis-py

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

OSS

Foundational Data Structures

4 redisinsight redisOM Java Jedis node-redis redis-py

Developer Experience

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

OSS

Foundational Data Structures

Real-Time Data Platform

4 redisinsight redisOM Java Jedis node-redis redis-py

Developer Experience

3 Search Functions

Redis-side Computing

STACK

2 JSON Probabilistic

Documents and Probabilistic Data Structures

OSS

1 Sets Hashes Sorted Sets Bit Field HyperLogLog

Strings Lists Bitmaps Geospatial Streams

Foundational Data Structures

A Quick Tour of Redis

The Data Balance

Structured vs. Unstructured

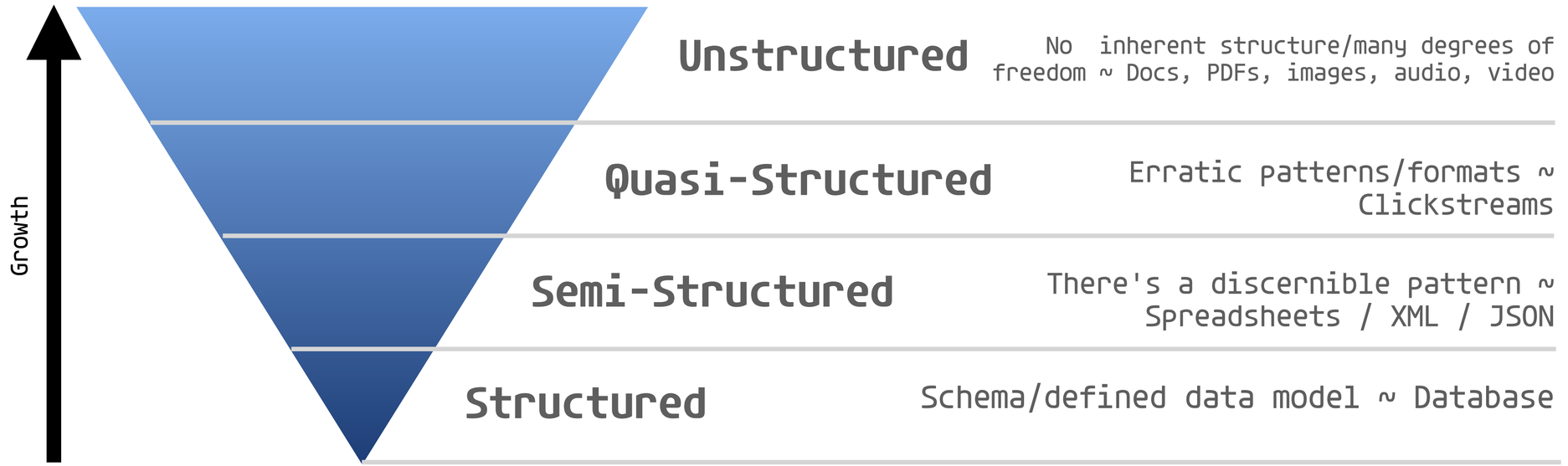
The balanced of data has changed radically...

~80% of the data generated by organizations is Unstructured

IDC report, 2020

... and this percentage is estimated to keep growing

with a compound annual growth rate (CAGR) of 36.5% between 2020 and 2025


Growth
Unstructured
No inherent structure/many degrees of freedom ~ Docs, PDFs, images, audio, video
Quasi-Structured
Erratic patterns/formats ~ Clickstreams
Semi-Structured
There's a discernible pattern ~ Spreadsheets / XML / JSON
Structured
Schema/defined data model ~ Database

How to deal with unstructured data?

Common approaches were labeling and tagging

These are labor intensive, subjective, and error-prone

Embeddings

Machine Learning Embeddings

Machine Learning/Deep Learning has leaped forward in last decade

ML models outperform humans in many tasks nowadays

🔥 CV (Computer Vision) models excel at detection/classification

🔥 LLMs (Large Language Models) have advanced exponentially

Machine Learning/Deep Learning has leaped forward in last decade

ML models outperform humans in many tasks nowadays

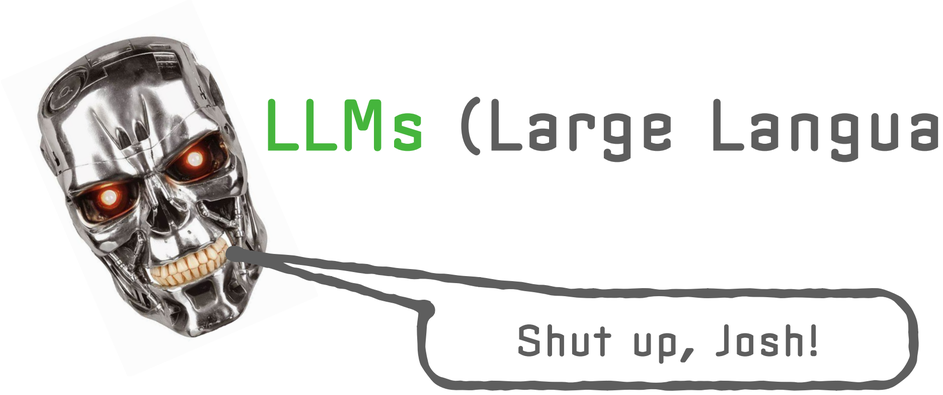
🔥 CV (Computer Vision) models excel at detection/classification

LLMs (Large Language Models) have advanced exponentially

Machine Learning/Deep Learning has leaped forward in last decade

ML models outperform humans in many tasks nowadays

🔥 CV (Computer Vision) models excel at detection/classification



LLMs (Large Language Models) have advanced exponentially

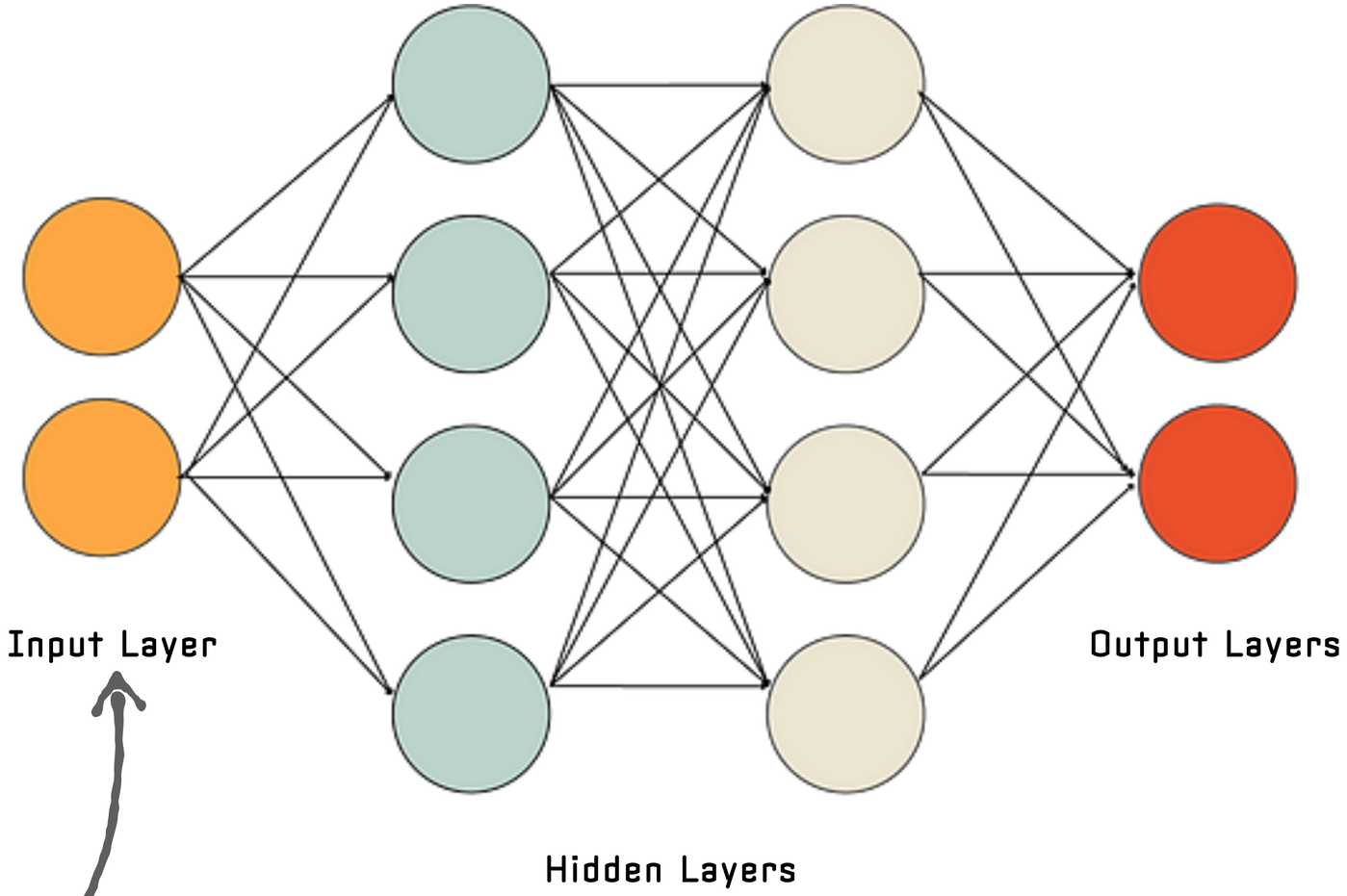
Feature Engineering

Raw Data

Feature	Value
Price	150
Category	Shirt

Scaled and 1-Hot Encoded

Feature	Value
Price	0.45
Category Shirt	1
Category Pants	0
Category Coats	0
Category Shoes	0



Shirt

Jacket

Automated Feature Engineering

ML models extract latent features

ML models embeddings catch the gray areas between features

The process of generating the embeddings is vectorizing

Vectorizing

Generating Vector Embedding for your Data

Vectorizing

Steps to Vectorizing

1. Choose an **Embedding Method**
2. **Clean** and **preprocess** the **data** as needed
3. **Train** the embedding model
4. Generate **Embeddings**

Vectorizing

Better Models, better Vectors

Embeddings can capture the semantics of complex data

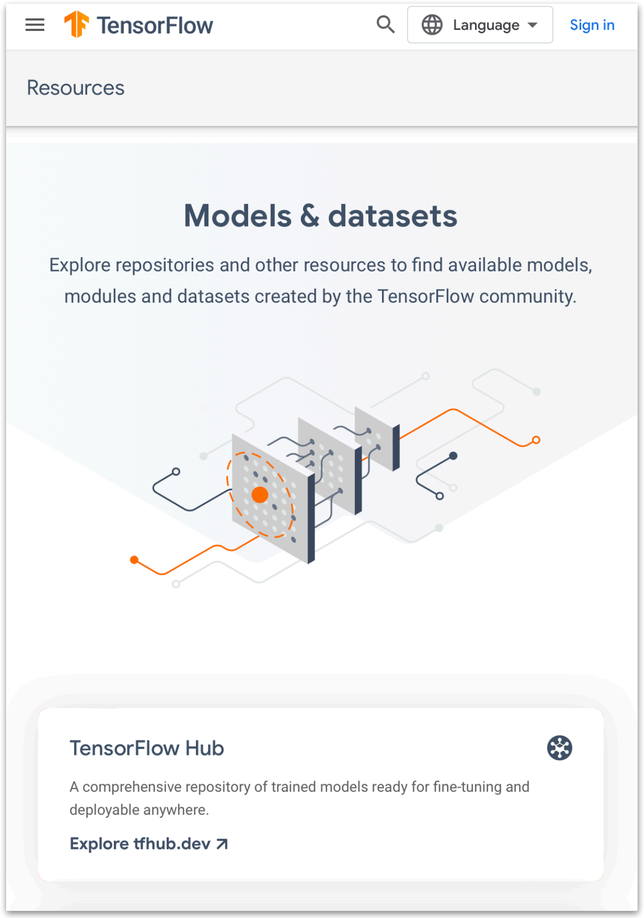
Option #1: Use a pre-trained model

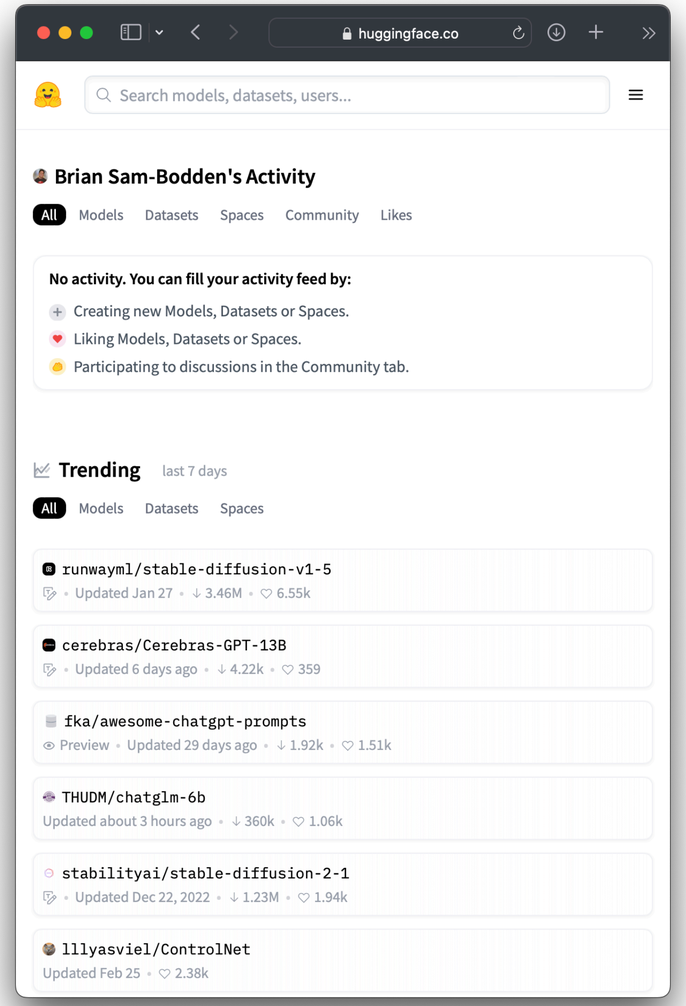
Option #2: train your models with custom data

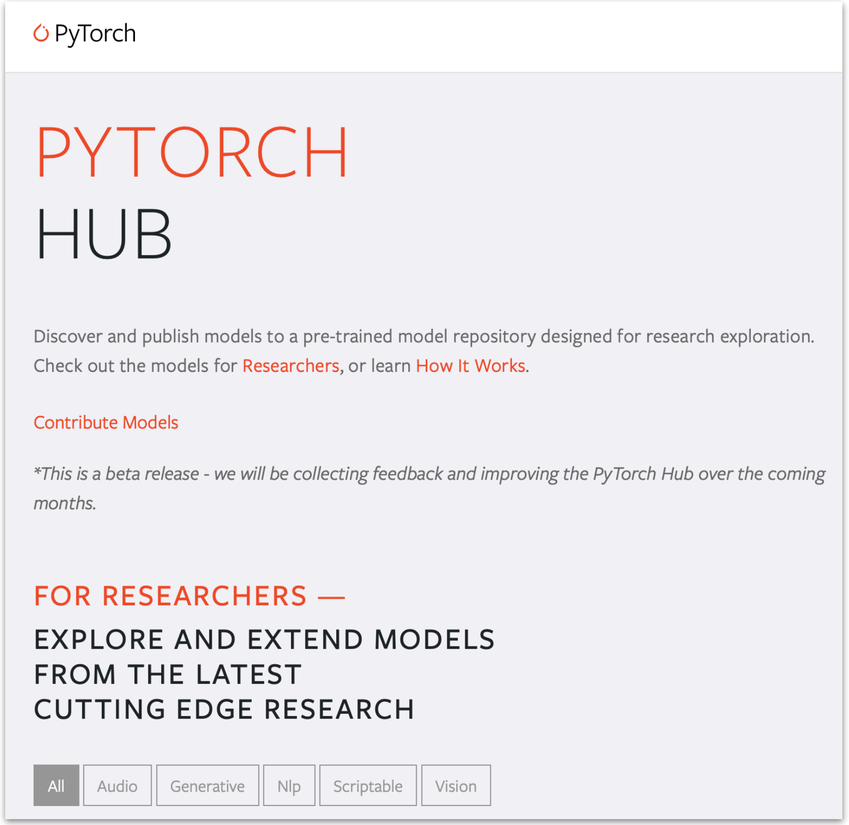
Vector similarity is a downline tool to analyze embeddings

Vectorizing

Ever growing collection of pre-trained, trainable and scriptable models

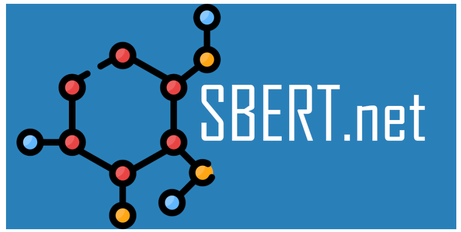






Vectorizing

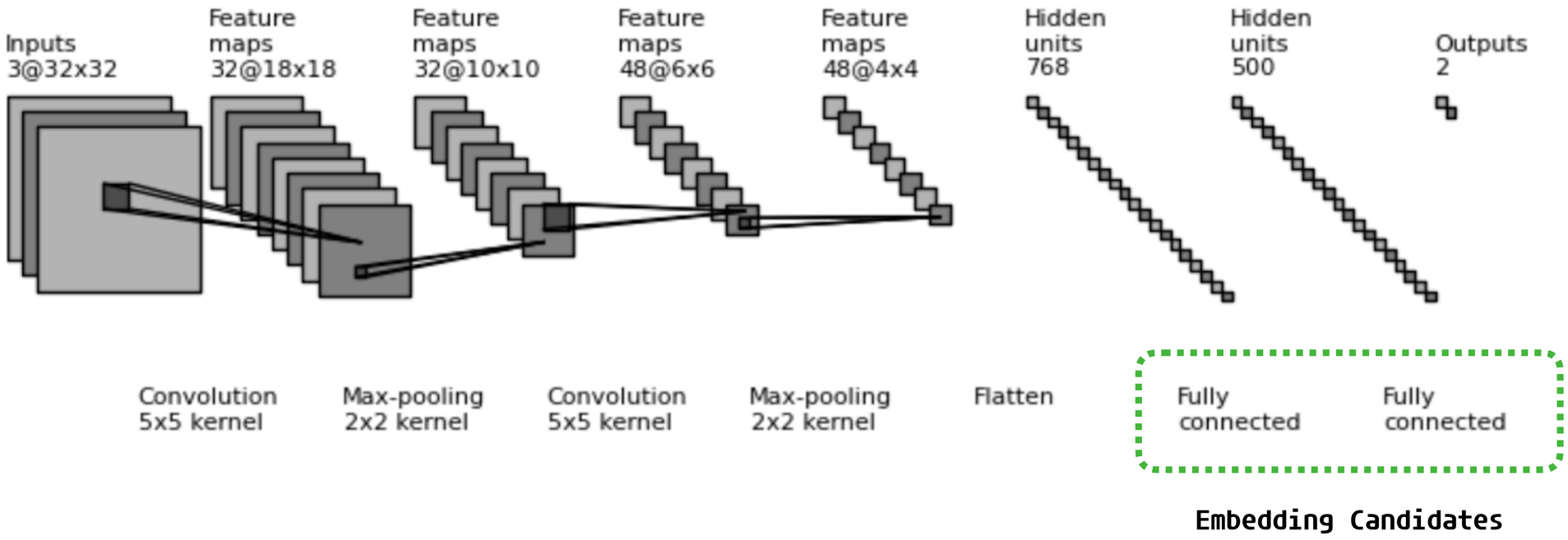
For sentences, SBERT.net provides a variety of pre-trained models:



Model Name	Performance Sentence Embeddings (14 Datasets)	Performance Semantic Search (6 Datasets)	Avg. Performance	Speed	Model Size
all-mpnet-base-v2	69.57	57.02	63.30	2800	420 MB
multi-qa-mpnet-base-dot-v1	66.76	57.60	62.18	2800	420 MB
all-distilroberta-v1	68.73	50.94	59.84	4000	290 MB
all-MiniLM-L12-v2	68.70	50.82	59.76	7500	120 MB
multi-qa-distilbert-cos-v1	65.98	52.83	59.41	4000	250 MB
all-MiniLM-L6-v2	68.06	49.54	58.80	14200	80 MB
multi-qa-MiniLM-L6-cos-v1	64.33	51.83	58.08	14200	80 MB
paraphrase-multilingual-mpnet-base-v2	65.83	41.68	53.75	2500	970 MB
paraphrase-albert-small-v2	64.46	40.04	52.25	5000	43 MB
paraphrase-multilingual-MiniLM-L12-v2	64.25	39.19	51.72	7500	420 MB
paraphrase-MiniLM-L3-v2	62.29	39.19	50.74	19000	61 MB
distiluse-base-multilingual-cased-v1	61.30	29.87	45.59	4000	480 MB
distiluse-base-multilingual-cased-v2	60.18	27.35	43.77	4000	480 MB

Vectorizing

Extracted a 1-dimensional layer that's densely packed with information about present features



Vectors

Storing and creating Vectors

Vectors

What's is a Vector?

Numeric representation of something in N-dimensional space

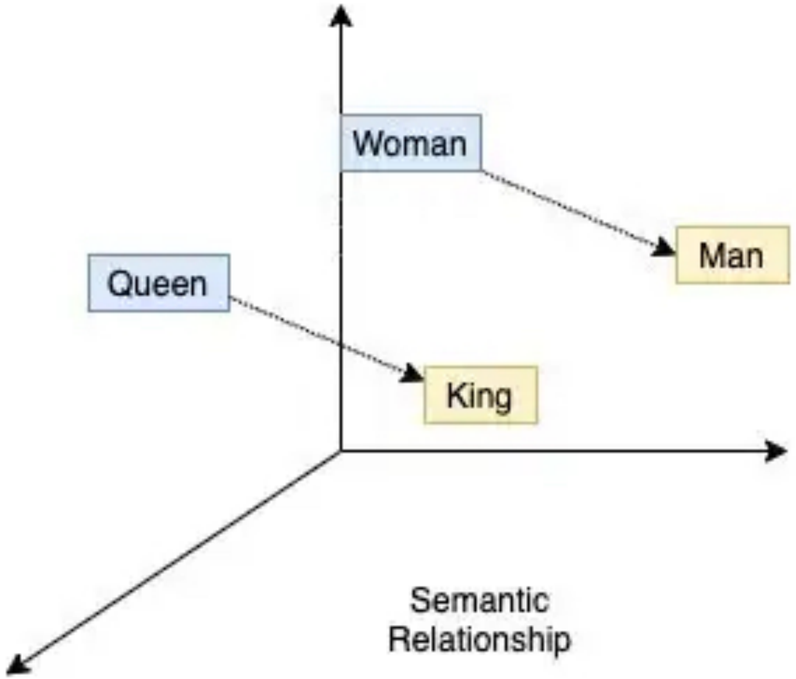
Can represent anything... entire documents, images, video, audio

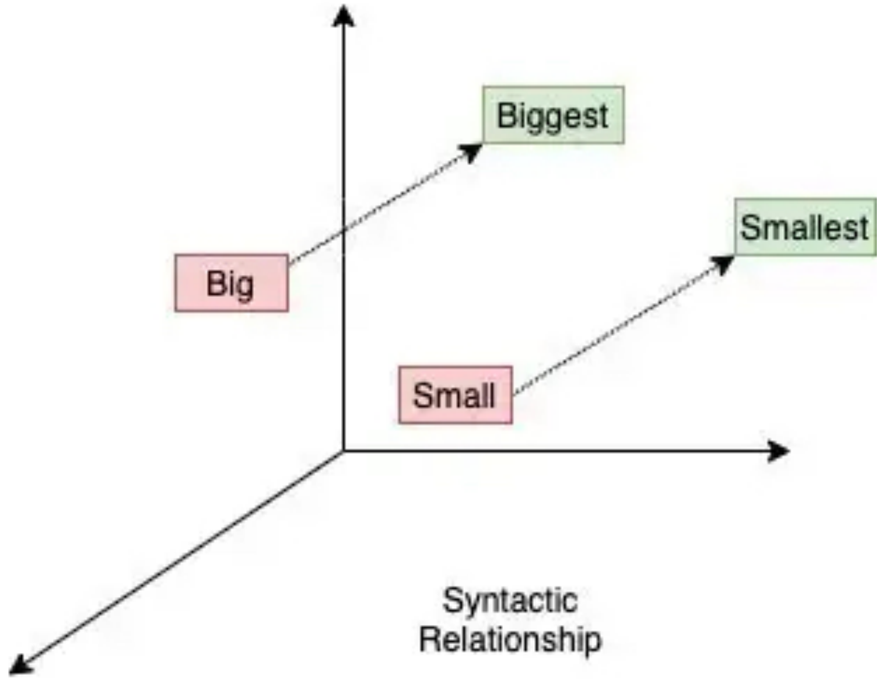
Quantifies features or characteristics of the item

More importantly... they are comparable

Vectors

Visually

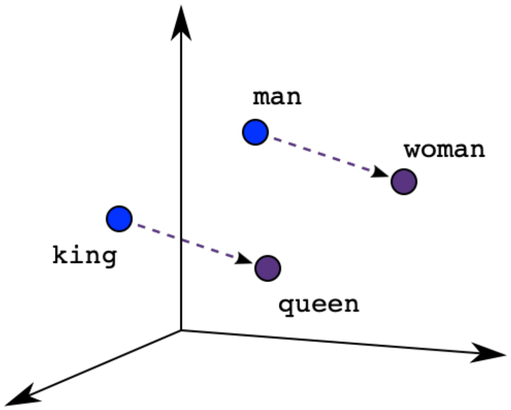




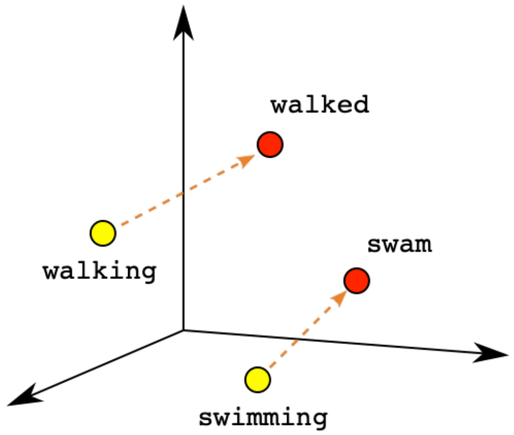
Vectors

Visually

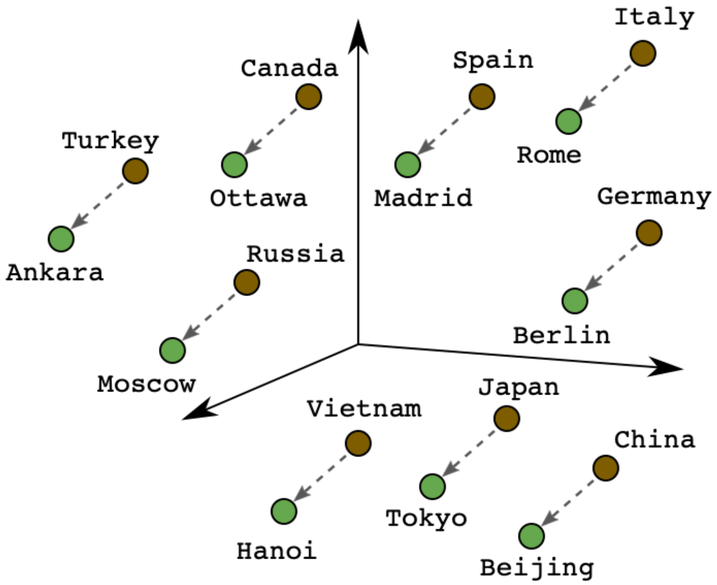
Male-Female



Verb Tense



Country-Capital



Vectors

Visually

“king”

“Man”

“Woman”

https://jalammar.github.io/illustrated-word2vec

Let's make some Vectors...

DJL

Machine Learning in Java

with DJL - https://djl.ai

A Java Framework for Machine Learning

Build, train, deploy ML and DL models

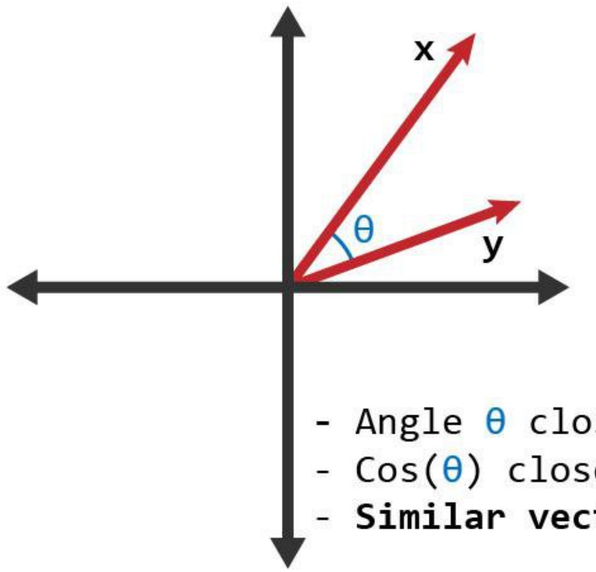
Abstracts PyTorch, Apache MXNet, TensorFlow and ONNX

Opens the world of "Model Zoos" to the Java Community

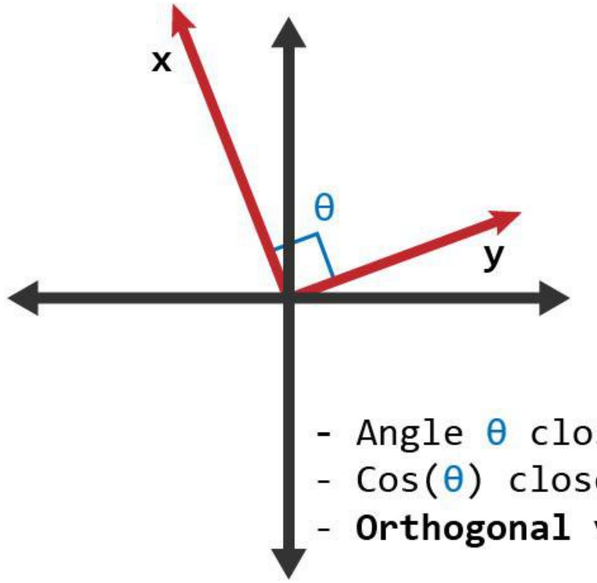
Similarity

Finding similar vectors

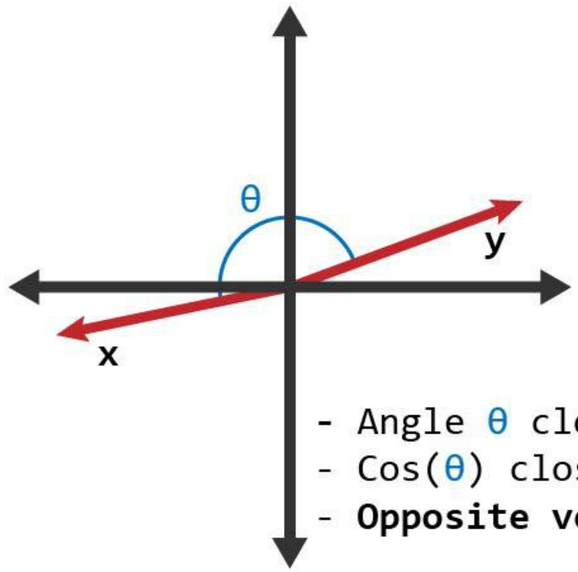
Measuring Similarity with Cosines



- Angle θ close to 0
- Cos(θ) close to 1
- **Similar vectors**

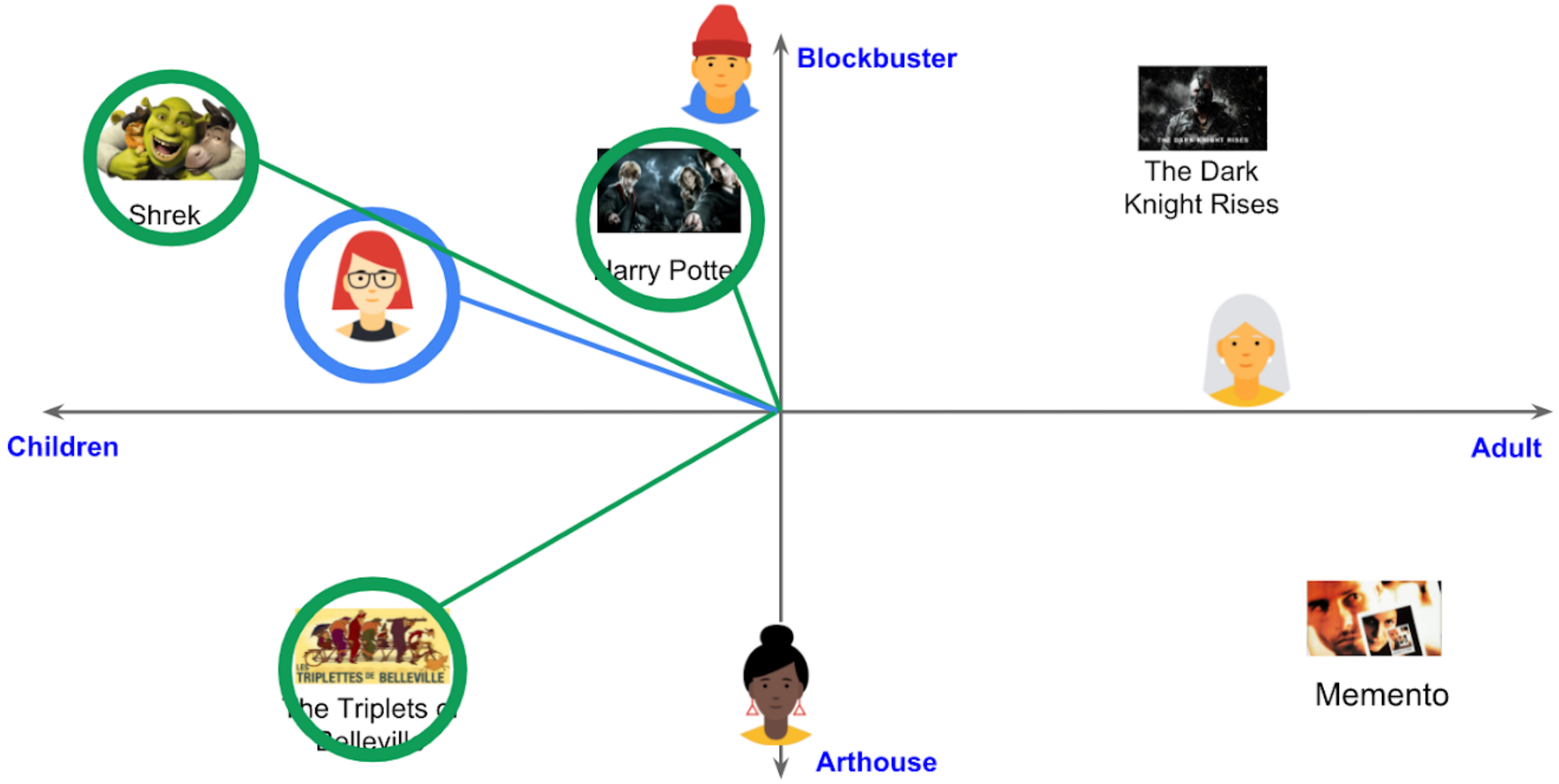


- Angle θ close to 90
- Cos(θ) close to 0
- **Orthogonal vectors**



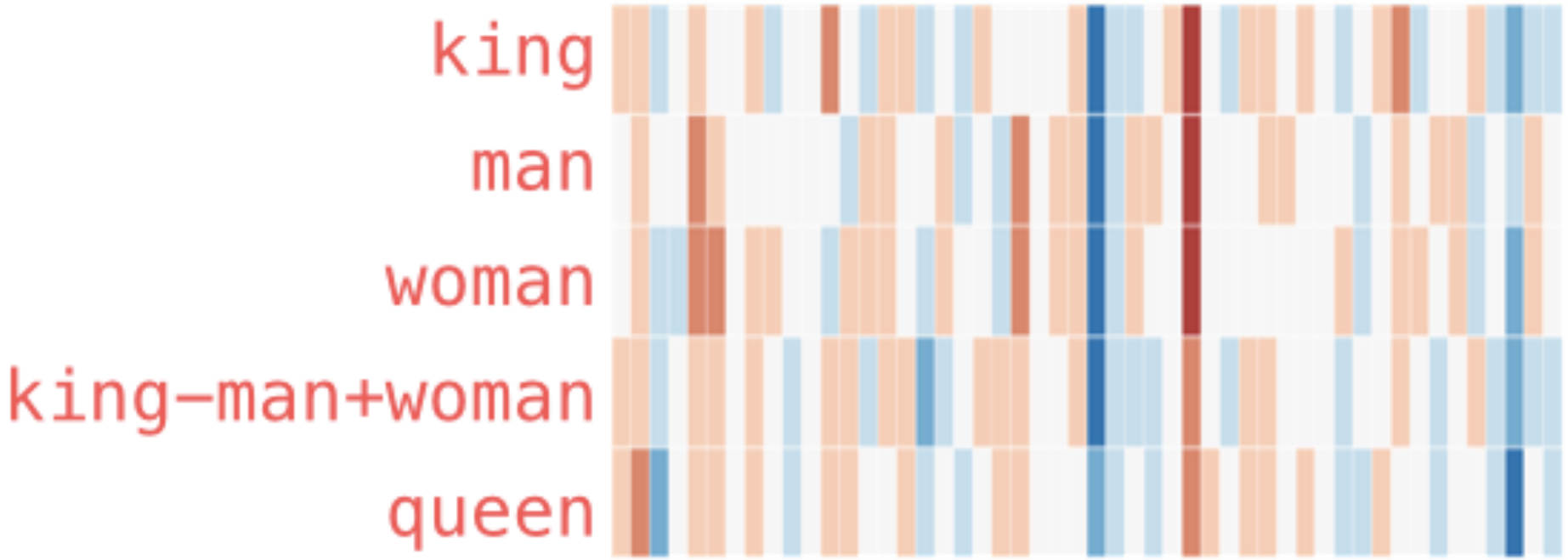
- Angle θ close to 180
- Cos(θ) close to -1
- **Opposite vectors**

An example of a 2D embedding space for Movies



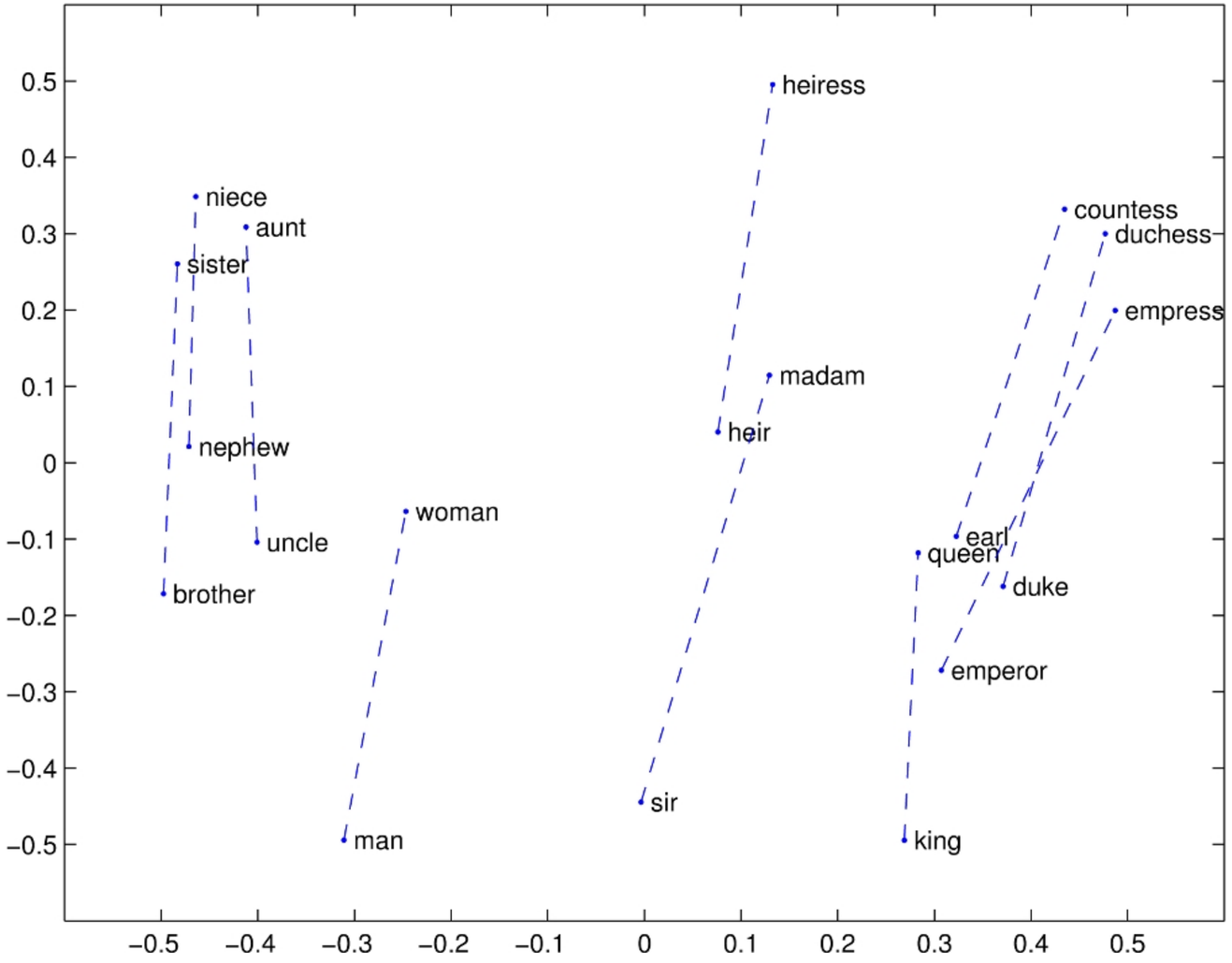
Vectors can be operated upon

king - man + woman ~= queen



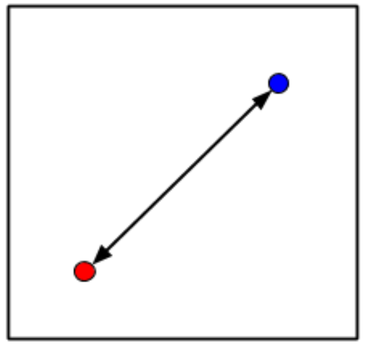
https://jalammar.github.io/illustrated-word2vec

Vector Difference between Pairs of Word Vectors

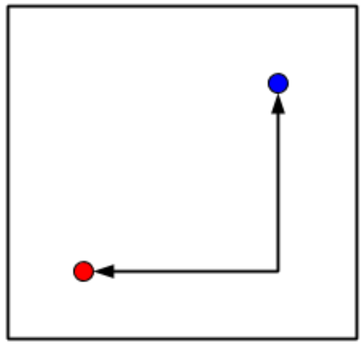


A few of the many Distance Metrics

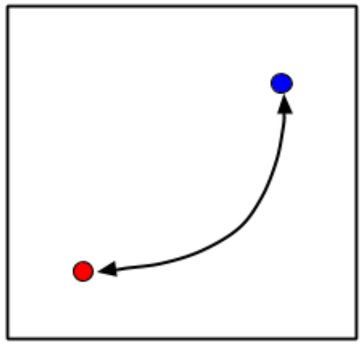
Euclidean



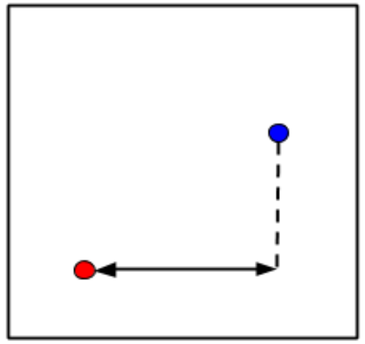
Manhattan



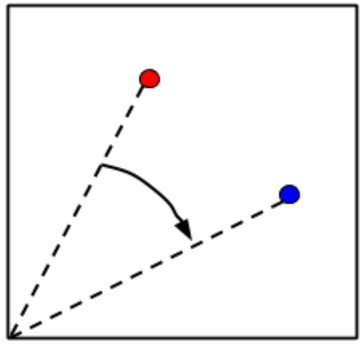
Minkowski



Chebychev



Cosine Similarity



Hamming

Redis OM Spring

Extending Spring Data Redis

Multi-Model Object-Mapping & Querying

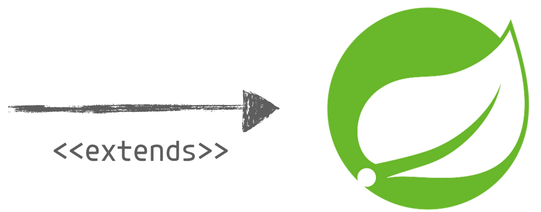
Multi-Model Object-Mapping & Querying

Redis OM Spring

Spring Data Redis

Multi-Model Object-Mapping & Querying

Redis OM Spring



Spring Data Redis

Multi-Model Object-Mapping & Querying

Jedis

redisOM

Redis OM Spring

<<extends>>

Spring Data Redis

Multi-Model Object-Mapping & Querying

Jedis

<<uses>>

redisOM

Redis OM Spring

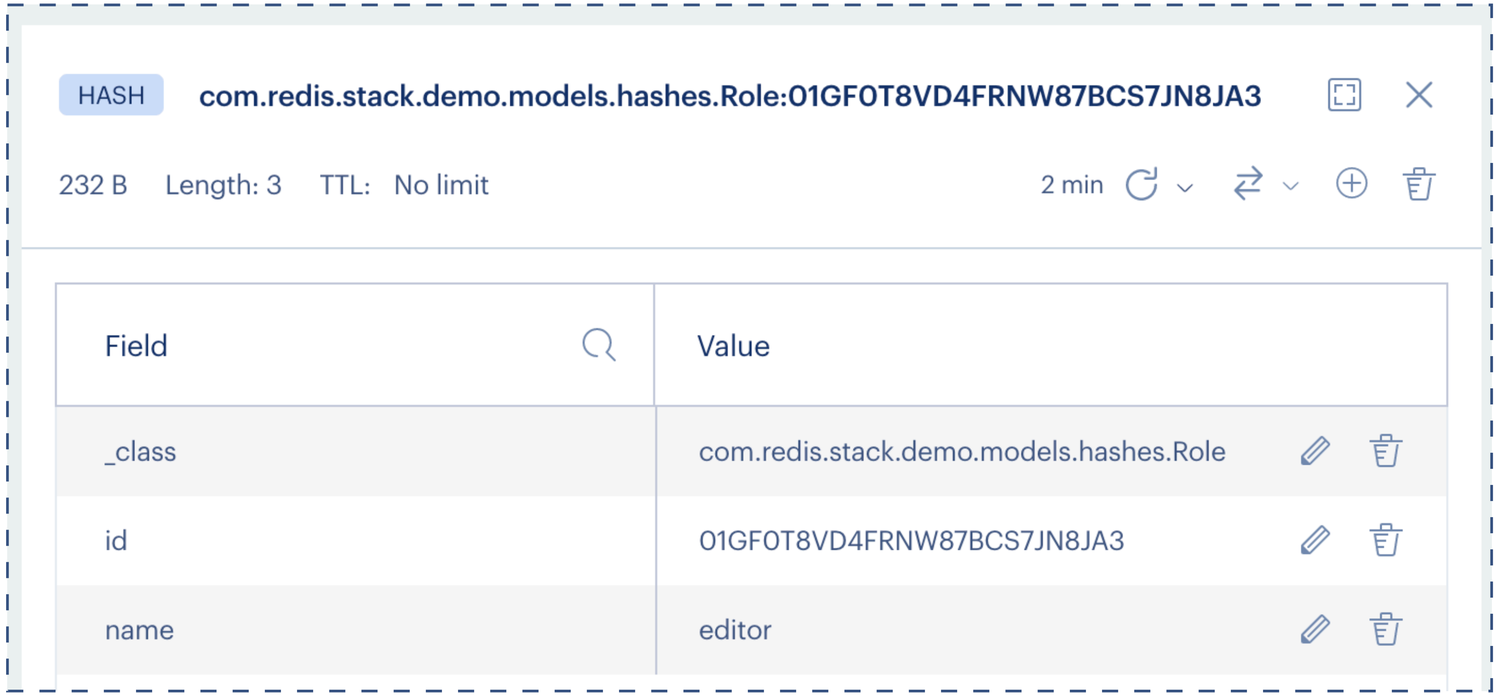
<<extends>>

Spring Data Redis

Model -> Redis Hash

@RedisHash Annotation

```
@Data
@Builder
@RedisHash
public class Role {
  @Id
  private String id;

  private String name;
}
```



Spring Data Repositories

From Spring Data Redis

```
@Repository
public interface RoleRepository
        extends CrudRepository<Role, String> {
}
```

Method and Description
`count()` Returns the number of entities available.
`delete(T entity)` Deletes a given entity.
`deleteAll()` Deletes all entities managed by the repository.
`deleteAll(Iterable<? extends T> entities)` Deletes the given entities.
`deleteAllById(Iterable<? extends ID> ids)` Deletes all instances of the type T with the given IDs.
`deleteById(ID id)` Deletes the entity with the given id.
`existsById(ID id)` Returns whether an entity with the given id exists.
`findAll()` Returns all instances of the type.
`findAllById(Iterable<ID> ids)` Returns all instances of the type T with the given IDs.
`findById(ID id)` Retrieves an entity by its id.
`save(S entity)` Saves a given entity.
`saveAll(Iterable<S> entities)` Saves all given entities.

Search Index Generation

@Indexed Annotation

```java
@Data
@Builder
@RedisHash
public class Role {
  @Id
  private String id;

  @Indexed
  private String name;
}
```

```java
@Repository
public interface RoleRepository
  extends CrudRepository<Role, String> {

  Optional<Role> findFirstByName(String name);
}
```

More Complex & Searchable Models

@Searchable / @Indexed / @Bloom Annotation

```
@Data
@RedisHash
public class User {
  @Id private String id;

  @Searchable private String name;

  @Bloom(name = "bf_company_email", capacity = 100000, errorRate = 0.001)
  @Indexed
  private String email;
  private String password;

  @Transient private String passwordConfirm;
  @Reference private Set<Role> roles;

  // audit fields
  @CreatedDate private Date createdDate;
  @LastModifiedDate private Date lastModifiedDate;
}
```

More Complex & Searchable Models

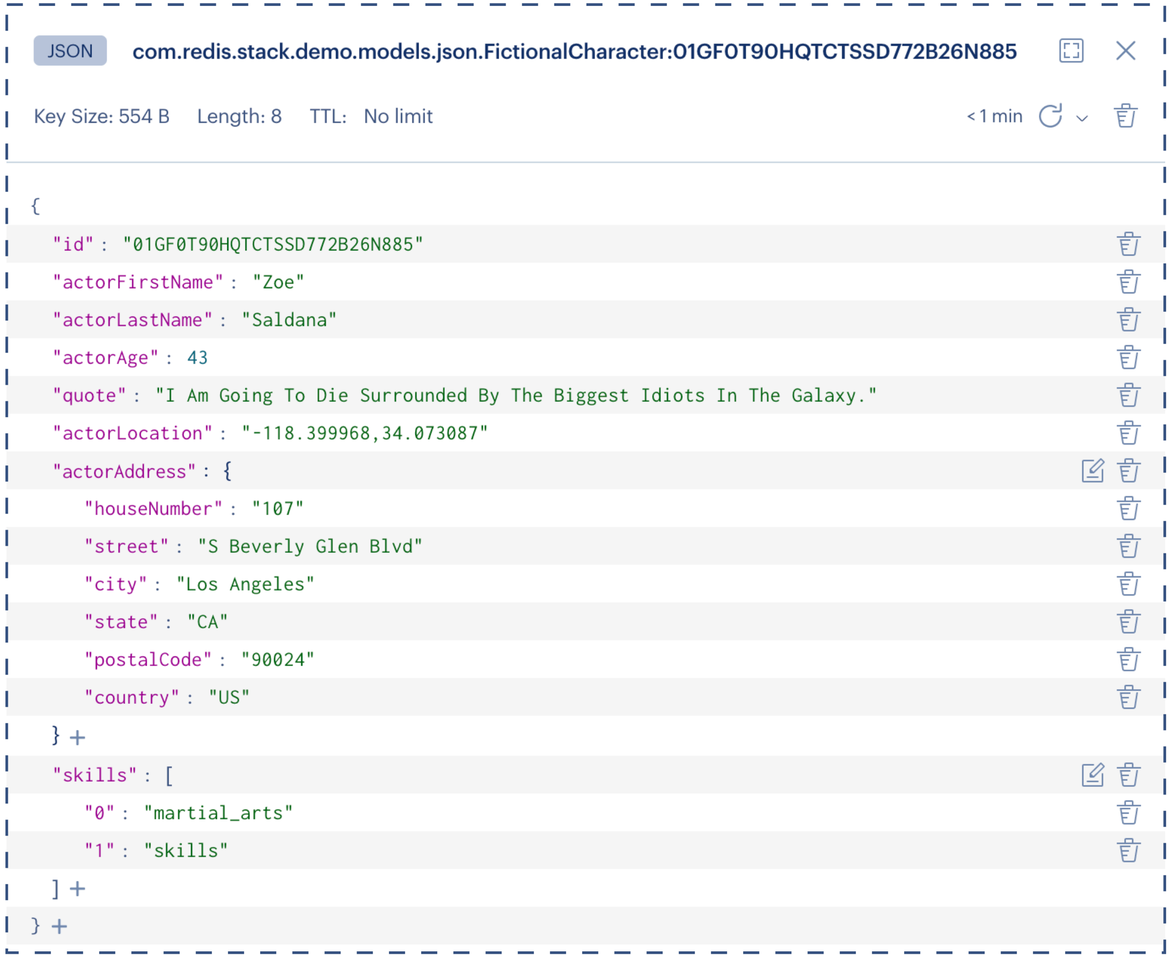
@Searchable / @Indexed / @Bloom Annotation

```
@Repository
public interface UserRepository extends CrudRepository<User, String> {
  Iterable<User> findByNameStartingWith(String prefix);

  Optional<User> findFirstByEmail(String email);

  boolean existsByEmail(String email);
}
```

Model -> Redis JSON

@Document Annotation

```java
@Document
public class FictionalCharacter {
  @Id
  @Indexed private String id;

  // Indexed for exact text matching
  @Indexed private String actorFirstName;
  @Indexed private String actorLastName;

  // Indexed for numeric matches
  @Indexed private Integer actorAge;

  // Indexed for Full Text matches
  @Searchable private String quote;

  // Indexed for Geo Filtering
  @Indexed private Point actorLocation;

  // Nest indexed object
  @Indexed private Address actorAddress;

  @Indexed private Set<String> skills;
}
```



More Repository Superpowers w/ JSON

@Searchable / @Indexed

```
public interface FictionalCharacterRepository extends RedisDocumentRepository<FictionalCharacter, String> {
  // Find people by age range
  Iterable<FictionalCharacter> findByActorAgeBetween(int minAge, int maxAge);

  // Find people by their first and last name
  Iterable<FictionalCharacter> findByActorFirstNameAndActorLastName(String firstName, String lastName);

  // Draws a circular geofilter around a spot and returns all people in that
  // radius
  Iterable<FictionalCharacter> findByActorLocationNear(Point point, Distance distance);

  // Performs full text search on a characters quote
  Iterable<FictionalCharacter> searchByQuote(String text);

  // Performing a tag search on city
  Iterable<FictionalCharacter> findByActorAddress_City(String city);

  // Search Characters that have one of multiple skills (OR condition)
  Iterable<FictionalCharacter> findBySkills(Set<String> skills);

  // Search Characters that have all of the skills (AND condition):
  Iterable<FictionalCharacter> findBySkillsContainingAll(Set<String> skills);
}
```

Auto-Complete Out of the Box

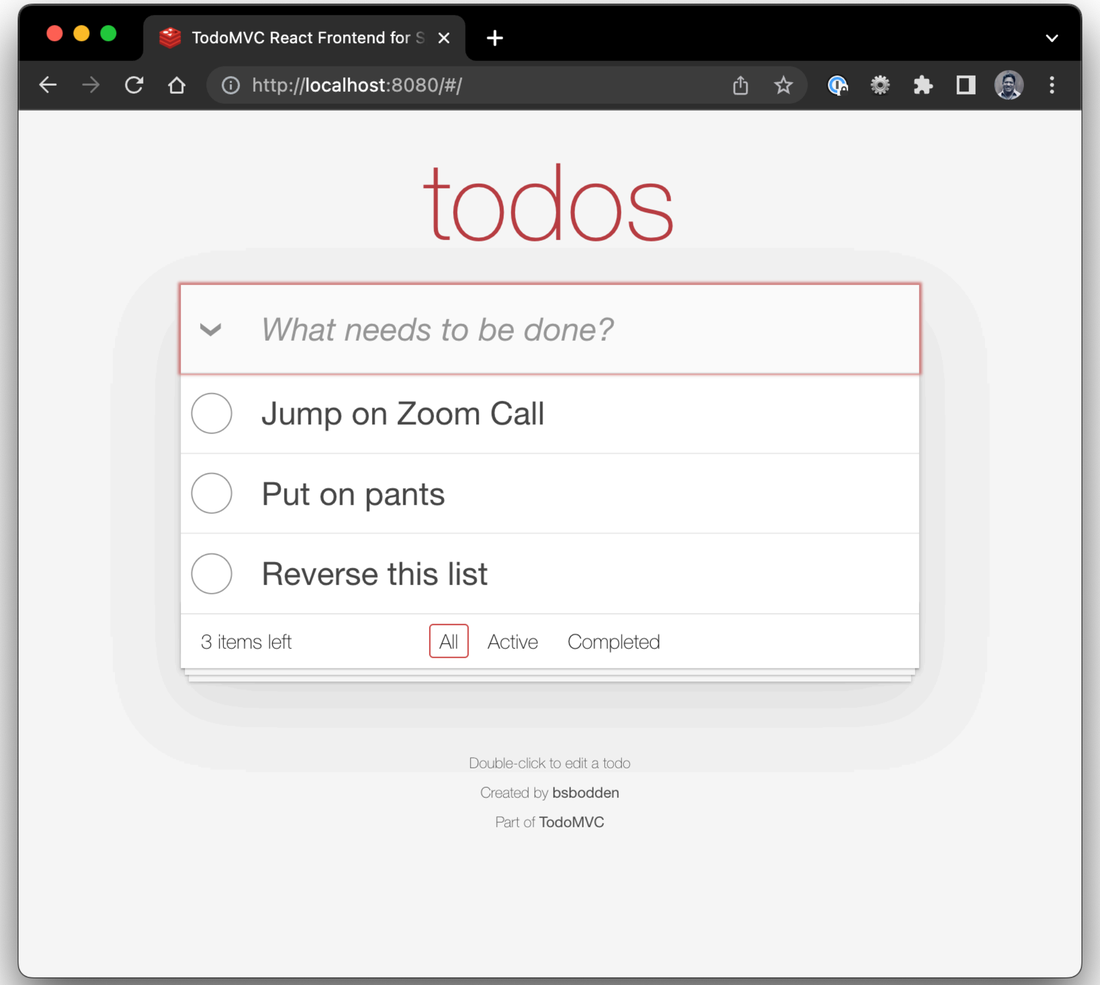
@AutoComplete / @AutoCompletePayload

```
@Document
public class CharacterEntry {
  @Id private String id;

  @AutoComplete @NonNull private String name;
  @AutoCompletePayload("name") private String type;

  @SerializedName("first appearance")
  @AutoCompletePayload("name") private String firstAppearance;
  @AutoCompletePayload("name") private Integer appearances;
}
```

```
public interface CharacterEntryRepository
       extends RedisDocumentRepository<CharacterEntry, String> {

  List<Suggestion> autoCompleteName(String query);
  List<Suggestion> autoCompleteName(String query, AutoCompleteOptions options);
}
```

Redis OM Todos

A Basic Redis OM Example



Searching...

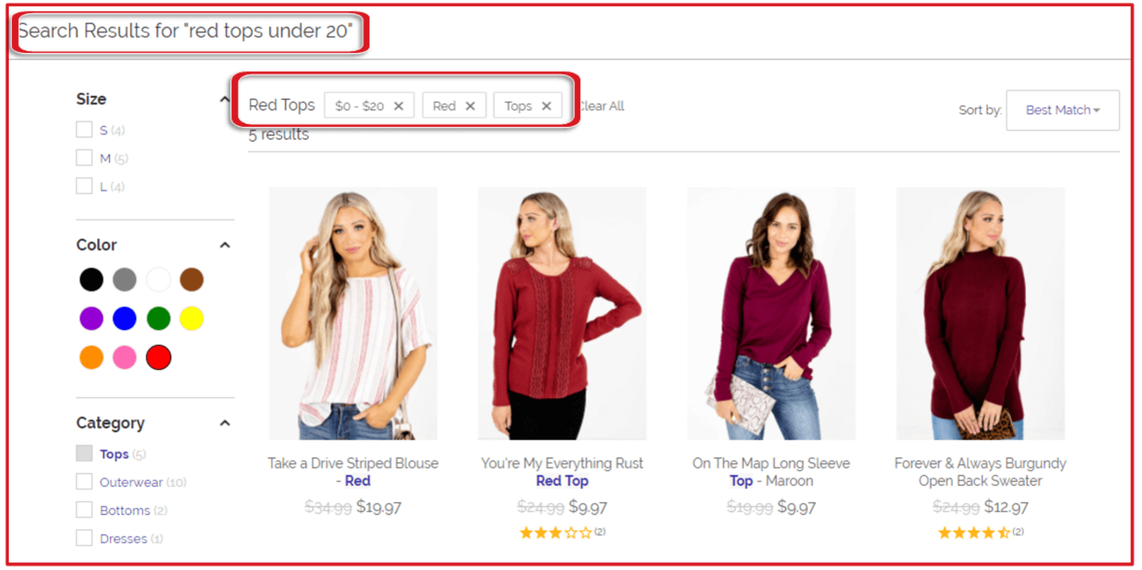
High Expectations

There's a growing expectation for effective search functionality

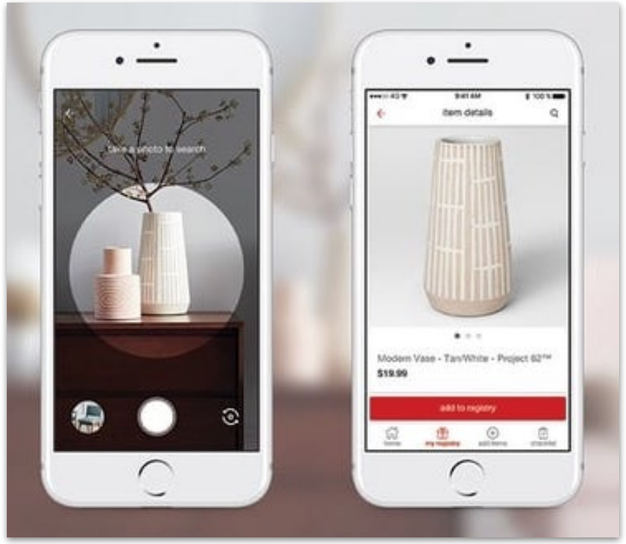
Unstructured Data is "high-dimensional"

Needs to capture "meaning and context" in unstructured data

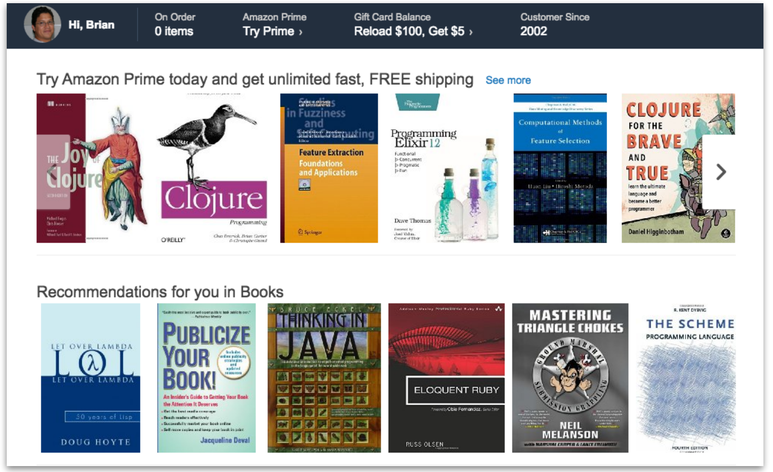
Encompasses Recommendation Engines and Similarity Search



Natural Language Search



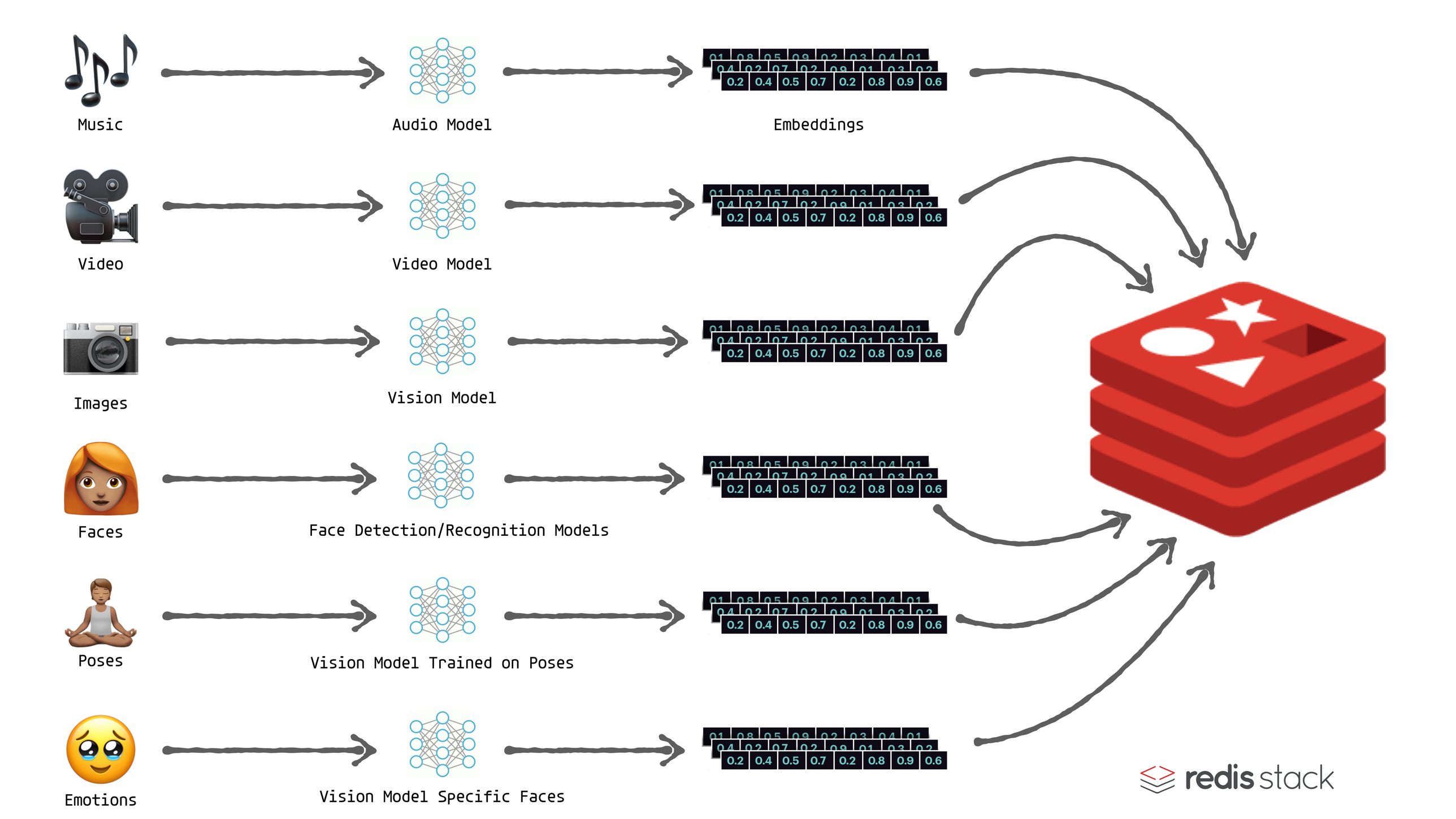
Visual Search



Recommenders

Vector Similarity in Redis

Redis as Vector Database


Music
Audio Model
Embeddings
Video
Video Model
Images
Vision Model
Faces
Face Detection/Recognition Models
Poses
Vision Model Trained on Poses
Emotions
Vision Model Specific Faces
0.2 0.4 0.5 0.7 0.2 0.8 0.9 0.6
redis stack

Vector Similarity in Redis

Index and query vector data stored as BLOBs in Redis Hashes/JSON

3 distance metrics: Euclidean, Internal Product and Cosine

2 indexing methods: HNSW and Flat

Hybrid queries combined with GEO, TAG, TEXT or NUMERIC

Celebrity Match Demo

Facial Similarity Search

Face Detection/Extraction

A peek under the hood

Let's look at the code to detect and extract faces...

Celebrity Match Demo

Breakdown

1. A **Celebrity** domain mapped to **Redis Hashes**
2. Spring Data **Repositories** powered by **RediSearch**
3. **@Indexed** annotated field for image embeddings
4. **@Vectorize** annotated field to generate embeddings
5. Upload image, detect/extract faces, gen. input embedding
6. Entity **Streams** to query for K nearest neighbors
7. Display **results**

@Indexed and @Vectorize

TLDR

The source of the embeddings and how to generate them

How to generate the Vector search schema field

```
@Vectorize(//
    destination = "imageEmbedding",
    embeddingType = EmbeddingType.FACE
)
@NonNull
private String imageResource;


@Indexed(//
    schemaFieldType = SchemaFieldType.VECTOR, //
    algorithm = VectorAlgo.HNSW, //
    type = VectorType.FLOAT32, //
    dimension = 512, //
    distanceMetric = DistanceMetric.L2, //
    initialCapacity = 10
)
private byte[] imageEmbedding;
```

Redis VSS Celebrities Demo

http://localhost:8080/#

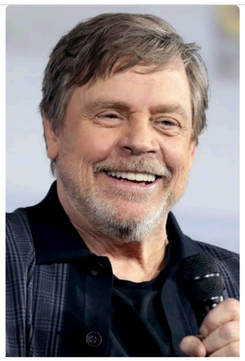
Redis VSS Celebs

Celebrity Match Demo

A Vector Similarity Search Example powered by Redis Stack.

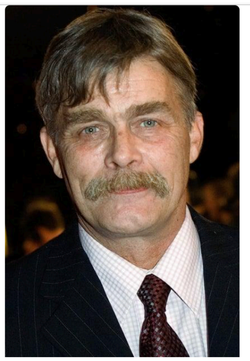
Choose File No file chosen Upload

New to Redis Stack? Visit the homepage and read about Vector Similarity Search.



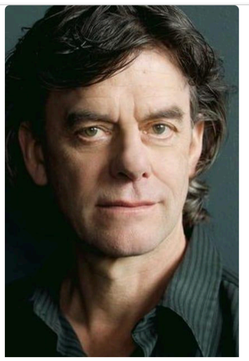
Mark Hamill

View 24.564



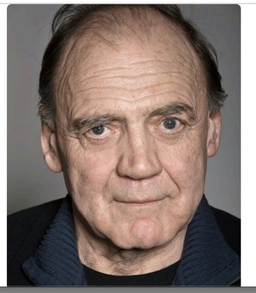
Nicholas Campbell

View 8.302



Tom McCamus

View 5.953



Fashion Product Finder

Image & Text - Hybrid Similarity Search

Redis VSS Demo

http://localhost:8080

Redis Vector Search Demo

About

Talk With Us

Fashion Product Finder

This demo uses the built-in Vector Search capabilities of Redis Stack to show how unstructured data, such as images and text, can be used to create powerful search engines.

Apply Filters | Load More Products

3000 searchable products

Rocky S Women White Handbag

View Similar:

By Text | By Image

Rocky S Women Blue Handbag

View Similar:

By Text | By Image

Rocky S Women Brown Handbag

View Similar:

By Text | By Image

Rocky S Women Red Clutch

View Similar:

By Text | By Image

Puma Unisex White Cap

View Similar:

By Text | By Image

The tools and techniques to unlock the value in Unstructured Data have evolved greatly...

Databases like Redis and frameworks like Redis OM Spring can help!

https://github.com/bsbodden/roms-vss-celebs

https://github.com/redis/redis-om-spring

https://redis.io/docs/stack/get-started/tutorials/stack-spring/

https://redis.com/blog/rediscover-redis-for-vector-similarity-search/

https://github.com/redis/jedis

Learn more at Redis Developer

https://developer.redis.com

Learn more at Redis University

https://university.redis.com

Thank you!!!

Thank you!!!