Face recognition and Postgres

Kobus Wolvaardt

+ Engineering and AI background
+ Medical software company
+ Has too many kids and needs face recognition software to help classify them
+ I will keep this very short

Why face recognition

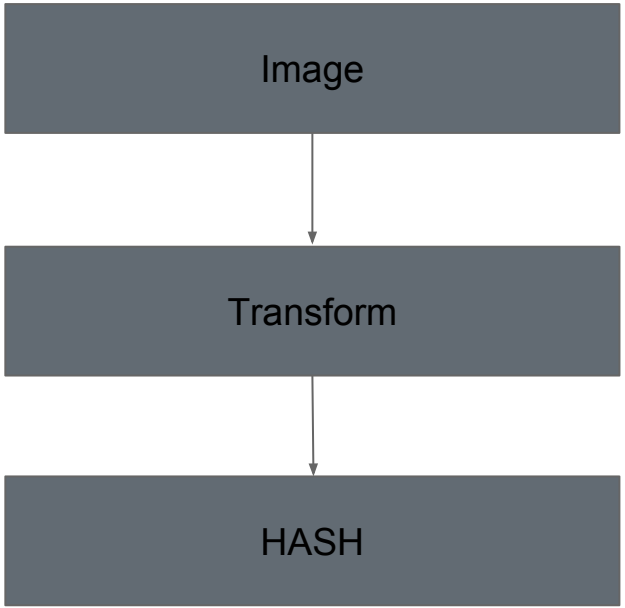
+ Cameras widely available
+ Software Libraries
+ Avatar common
+ Patient image help medical legal
+ Face based search convenient
+ It really demonstrates Postgres’s extensibility

Ethical issues

+ Get consent
+ Improve people’s lives if you can
+ Store only what you need
+ Get consent… really...

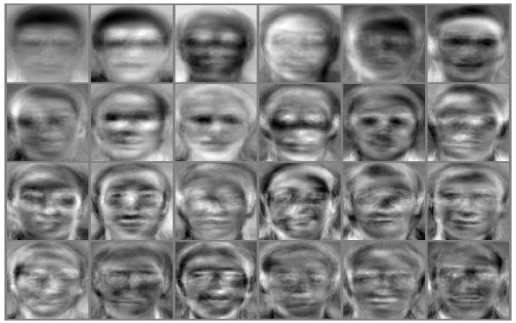
Face recognition

+ Geometric
+ Photometric

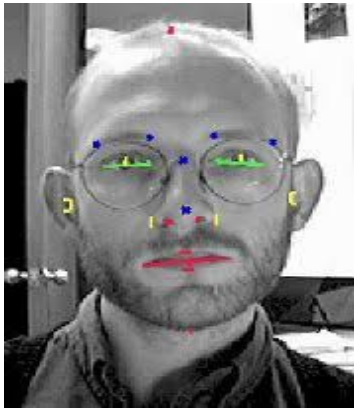


Earlier techniques

+ Photometric like Eigen faces

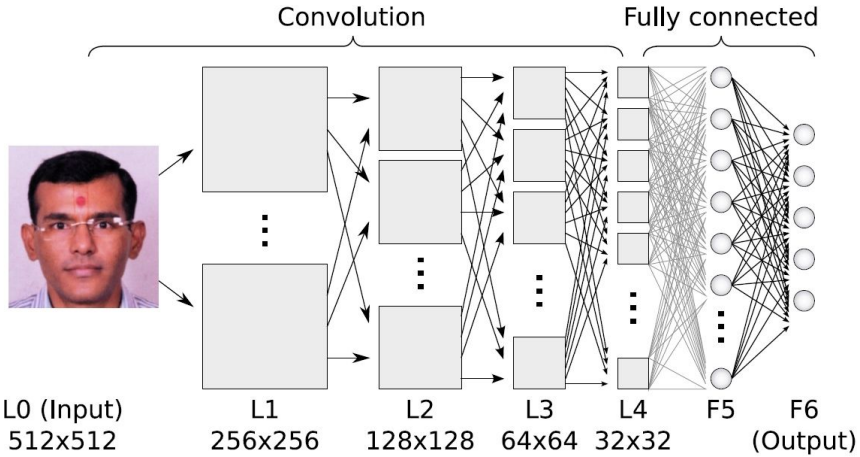


+ Geometric like Feature extraction



Deep learning

+ Deep learning can do unexpected things
+ Researchers trained a neural net to output a unique hash per face
+ Similar hash if same face



Postgres extendability (languages)

+ Python

Postgres extendability (languages)

+ Python
+ Perl

Postgres extendability (languages)

+ Python
+ Perl
+ SQL

Postgres extendability (languages)

+ Python
+ Perl
+ SQL
+ TCL

Postgres extendability (languages)

+ Python
+ Perl
+ SQL
+ TCL
+ JAVA
+ LUA
+ R
+ SH
+ Javascript

Postgres extendability (languages)

+ Python
+ Perl
+ SQL
+ TCL
+ JAVA
+ LUA
+ R
+ SH
+ Javascript
+ More non contrib ones

Postgres extendability (languages)

+ Python
+ Perl
+ SQL
+ TCL
+ JAVA
+ LUA
+ R
+ SH
+ Javascript
+ More non contrib ones
+ Can extend with custom modules

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)
+ Datetime (date, timestamp)

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)
+ Datetime (date, timestamp)
+ Documents stores (hstore, xml, json and jsonb)

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)
+ Datetime (date, timestamp)
+ Documents stores (hstore, xml, json and jsonb)
+ Enums

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)
+ Datetime (date, timestamp)
+ Documents stores (hstore, xml, json and jsonb)
+ Enums
+ Geo (line, circle, paths and more)
+ Network types
+ Range types

Postgres types

+ Numbers (int, numeric, money, and many levels of precision)
+ Strings (varchar, text, citext)
+ Datetime (date, timestamp)
+ Documents stores (hstore, xml, json and jsonb)
+ Enums
+ Geo (line, circle, paths and more)
+ Network types
+ Range types
+ Arrays and cubes
+ Custom types based on existing types.

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass
+ Bloom, btree_gin, btree_gist index

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass
+ Bloom, btree_gin, btree_gist index
+ Dblink, fdw and file_fdw

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass
+ Bloom, btree_gin, btree_gist index
+ Dblink, fdw and file_fdw
+ Earthdistance, fuzzystrmatch
+ Isn (types for barcodes)

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass
+ Bloom, btree_gin, btree_gist index
+ Dblink, fdw and file_fdw
+ Earthdistance, fuzzystrmatch
+ Isn (types for barcodes)
+ Pg_stat_statements and pg_buffer_cache

Postgres need even more?
Meet Contrib

+ Adminpack, auth_delay
+ Citext, cube, chkpass
+ Bloom, btree_gin, btree_gist index
+ Dblink, fdw and file_fdw
+ Earthdistance, fuzzystrmatch
+ Isn (types for barcodes)
+ Pg_stat_statements and pg_buffer_cache
+ Pg_trgm allows indexed partial string matches
+ Tablefunc with crosstab
+ Much much more

Our face implementation

+ ~~Cheating and stealing~~ Leverage other people's work
+ Python dlib wrapper
+ Postgres python wrapper
+ Postgres cube type
+ Two pg functions and a trigger
+ JS code to transport the photos

Postgres language extension

+ Postgres supports many languages
+ Python happens to be supported
+ CREATE LANGUAGE plpython3u;
+ Python trigger upon insert to calculate the face hash (fash)

Postgres type extension

Array’s with distance measure as face match, wasn’t there a module for that?

CREATE EXTENSION cube;

+ Cube provides distance queries
+ Cube provides indexed distance searches
+ <-> operator calculates distance and gist index allows indexed searching. 9.6 and later

Implementation

```
CREATE EXTENSION cube;


CREATE TABLE facetable
(
-- Person name
    name text,
-- Person image (jpg or png) in base64 from javascript canvas
    image text,
-- Person face hash
    fash cube
);
```

Implementation

```
CREATE OR REPLACE FUNCTION update_fash()
RETURNS trigger AS $BODY$
try:
    import base64, face_recognition, PIL
    try:
        data = base64.b64decode(TD["new"]["image"])
    except:
        data = TD["new"]["image"]
    Im = np.array(PIL.Image.frombytes(data))
    fash = face_recognition.face_encodings(im)[0]
    TD["new"]["fash"] = fash
    return "MODIFIED"
except:
    return "OK"
$BODY$ LANGUAGE plpython3u;

CREATE TRIGGER get_fash BEFORE UPDATE OR INSERT ON
facetable FOR EACH ROW EXECUTE PROCEDURE update_fash();
```

Lets have some fun

Visit: https://bit.ly/2INfao8 or https://face.quantsolutions.co.za:9443

Take a picture of your face and type your name, submit. You will be redirected to:

https://face.quantsolutions.co.za:9443/facefind

Now find some faces by Snap Photo with one or more faces in view.

This might not work on your browser (I am not a javascript whisperer)

Performance

We can search in 1 000 000 records on PG9.6 and my i5 desktop in about 1200ms

The power of PostgreSQL

- Proper programming languages
- Datatypes
- More datatypes
- Custom Extensions
- Did I mention datatypes?
- Any immutable function can be indexed
- GIN and GIST indexes for containment style queries
- I almost forgot to mention the wide array of datatypes

Questions

Questions?