algolia

Zucchinis or Cucumbers?

Benchmarking Embeddings
for **Similar Image Retrieval**
thanks to your **weekly Grocery shopping**

Let’s start with a little game

algolia
Powering Discovery

Looking similar

Zucchini_025.jpg_
test

Zucchini_001.jpg_
train

Zucchini_022.jpg_
test

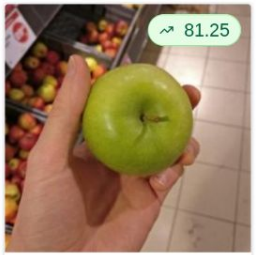
Zucchini_018.jpg_
train

Zucchini_007.jpg_
test

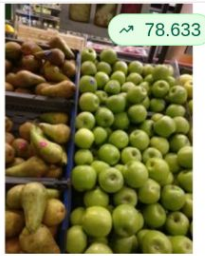
How many of these are *Cucumbers* ? 🥒

Let’s start with a little game

algolia
Powering Discovery



Mango_001.jpg_test



Pomegranate_019.
jpg_test

How many of these are *Apples*? 🍏

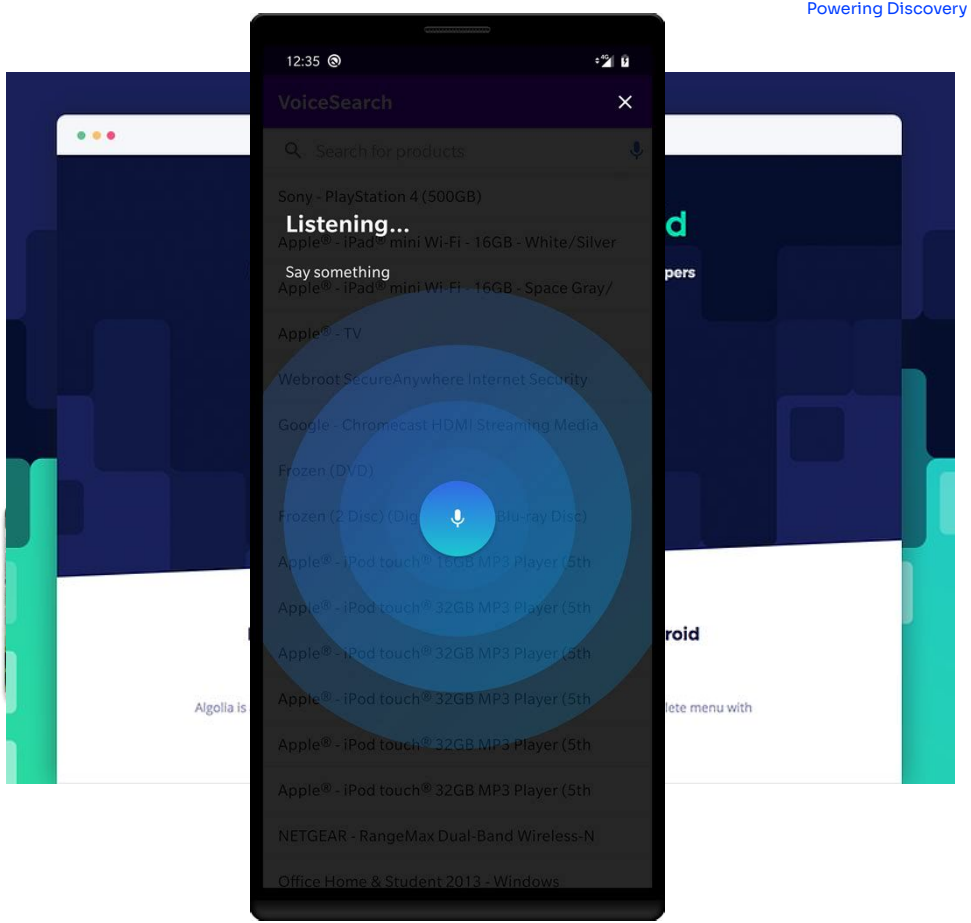
Hello, my name is Paul-Louis Nech

- **Coder** for more than 15 years
- **Software engineer** since 2016
- **Machine Learning** for the last 5 years

algolia
Powering Discovery

I build APIs & tools at algolia

- Android **UI libraries**
- Voice Search **tooling**
- Natural Language **APIs**
- Image Recommendation **engine**

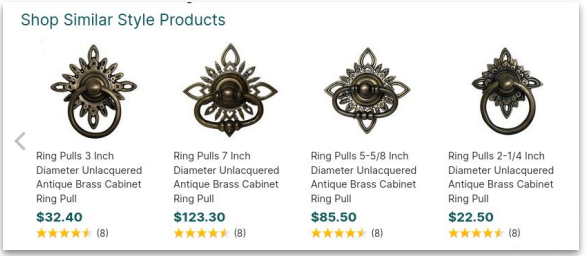


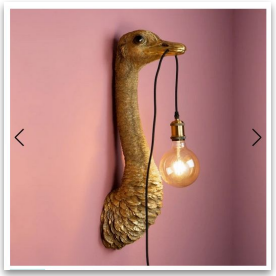
algolia
Powering Discovery

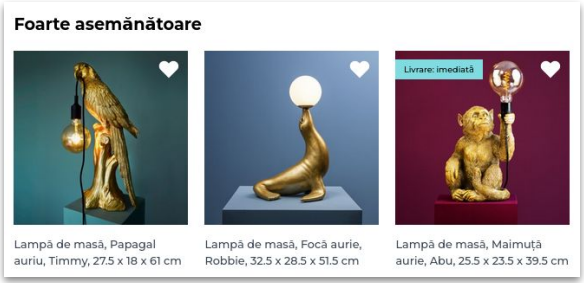
My last project: LookingSimilar

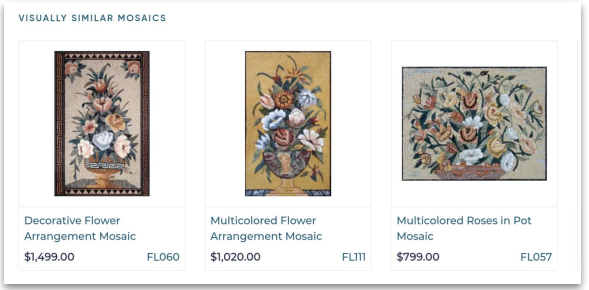
An **Image Recommendation API** based on **image embeddings** and **ANN retrieval**

CHECK IT OUT









LookingSimilar: an image vector retrieval API

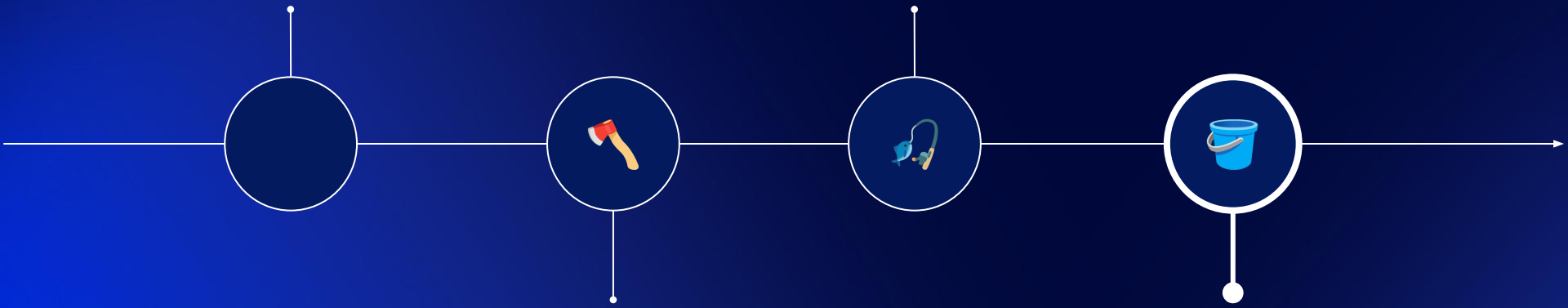
algolia
Powering discovery

Vectorization
With an image model

Hashing
Learned LSH

Retrieval
With HNSW

Filtering
Using object-level metadata



algolia
Powering Discovery

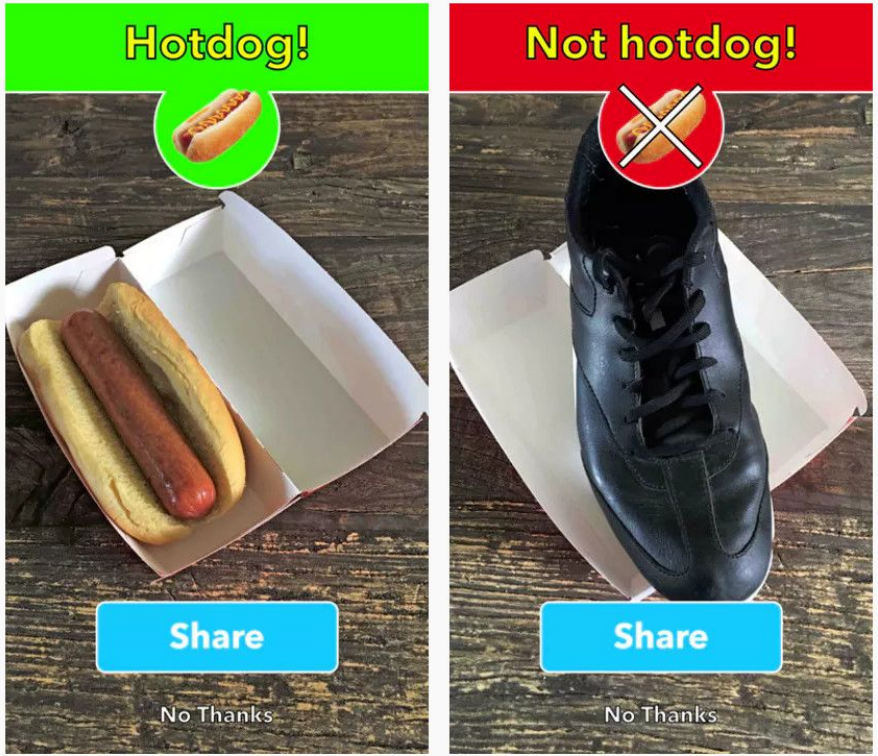
It all comes from a customer question

“Do you handle groceries well?”

algolia
Powering Discovery

This is a good question

Do we know how to recognize food?

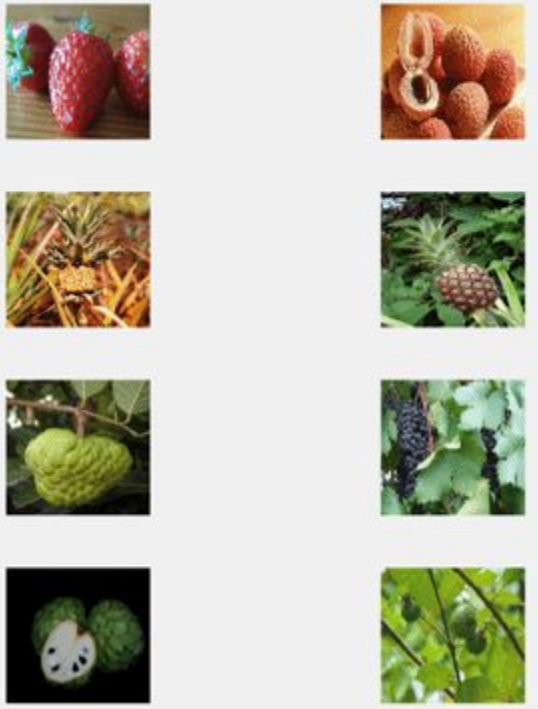


algolia
Powering Discovery

Searching for a Benchmark 🍏🥒

Surely there must be industry standards?

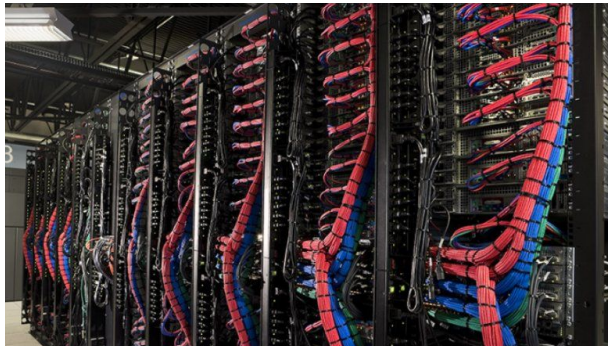
947	mushroom
948	Granny Smith
949	strawberry
950	orange
951	lemon
952	fig
953	pineapple, ananas
954	banana
955	jackfruit, jak, jack
956	custard apple
957	pomegranate



algolia
Powering Discovery

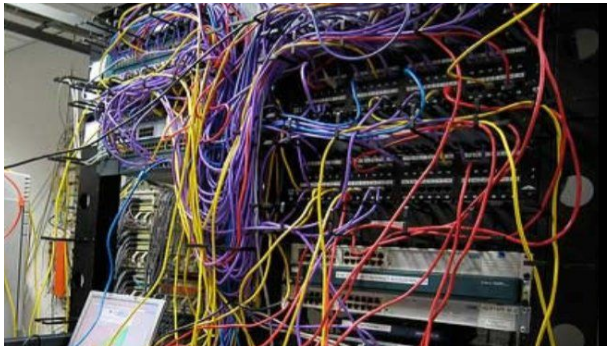
Expectation

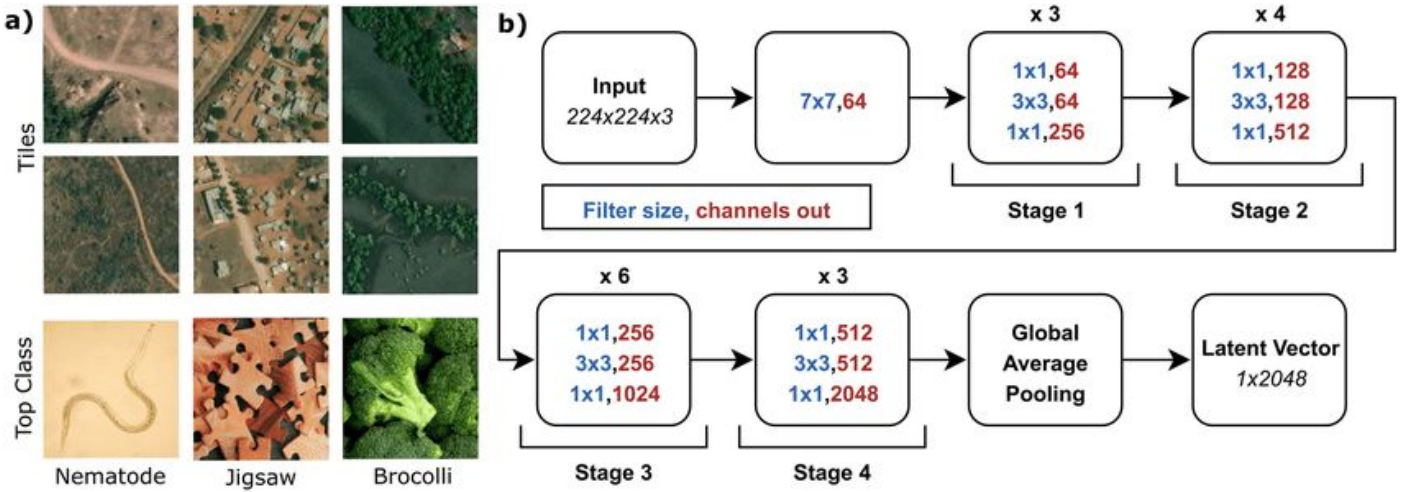
You can use **industry benchmarks** on your **SOTA** model



Reality

Industry benchmarks are **already used** in training, test, or validation -> probably **biased** in many **surprising ways**





ImageNet data -> ResNet training

You can't use the **training data** to test the model ⚠️

algolia
Powering Discovery

CLIP also is biased by ImageNet

If not **at training**,
definitely **tested**
and **iterated on**

-> You could say this dataset was **gamed against** by the OpenAI team

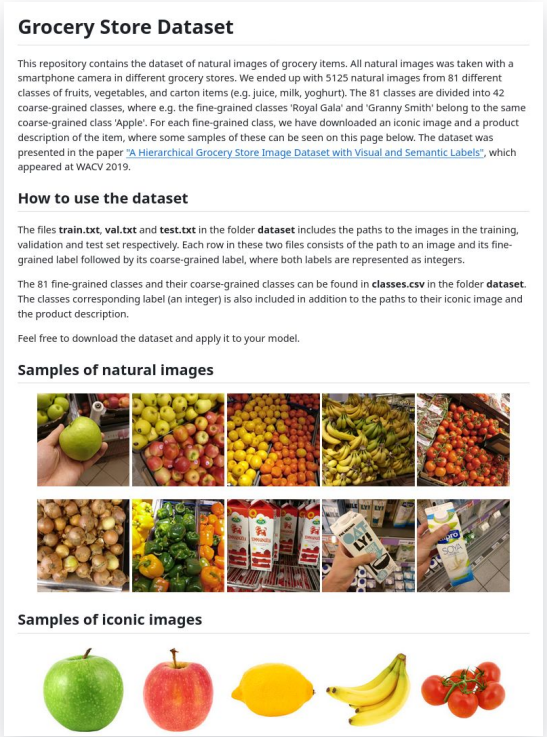
algolia
Powering Discovery

There is no good benchmark

But we still need to answer that question

Grocery Store dataset

algolia
Powering discovery



Grocery Store Dataset

This repository contains the dataset of natural images of grocery items. All natural images was taken with a smartphone camera in different grocery stores. We ended up with 5125 natural images from 81 different classes of fruits, vegetables, and carton items (e.g. juice, milk, yoghurt). The 81 classes are divided into 42 coarse-grained classes, where e.g. the fine-grained classes 'Royal Gala' and 'Granny Smith' belong to the same coarse-grained class 'Apple'. For each fine-grained class, we have downloaded an iconic image and a product description of the item, where some samples of these can be seen on this page below. The dataset was presented in the paper "A Hierarchical Grocery Store Image Dataset with Visual and Semantic Labels", which appeared at WACV 2019.

How to use the dataset

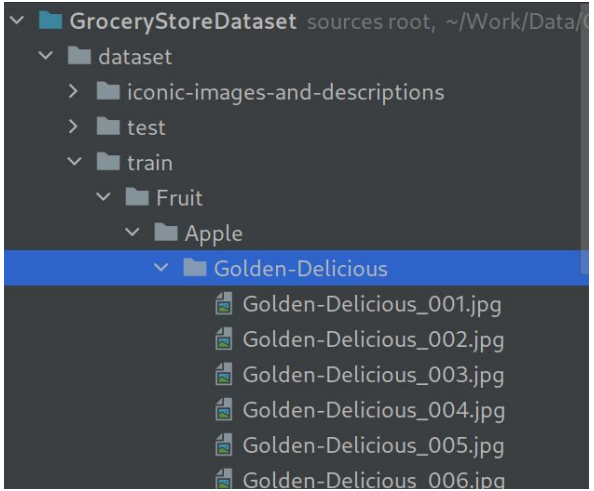
The files **train.txt**, **val.txt** and **test.txt** in the folder **dataset** includes the paths to the images in the training, validation and test set respectively. Each row in these two files consists of the path to an image and its fine-grained label followed by its coarse-grained label, where both labels are represented as integers.

The 81 fine-grained classes and their coarse-grained classes can be found in **classes.csv** in the folder **dataset**. The classes corresponding label (an integer) is also included in addition to the paths to their iconic image and the product description.

Feel free to download the dataset and apply it to your model.

Samples of natural images

Samples of iconic images

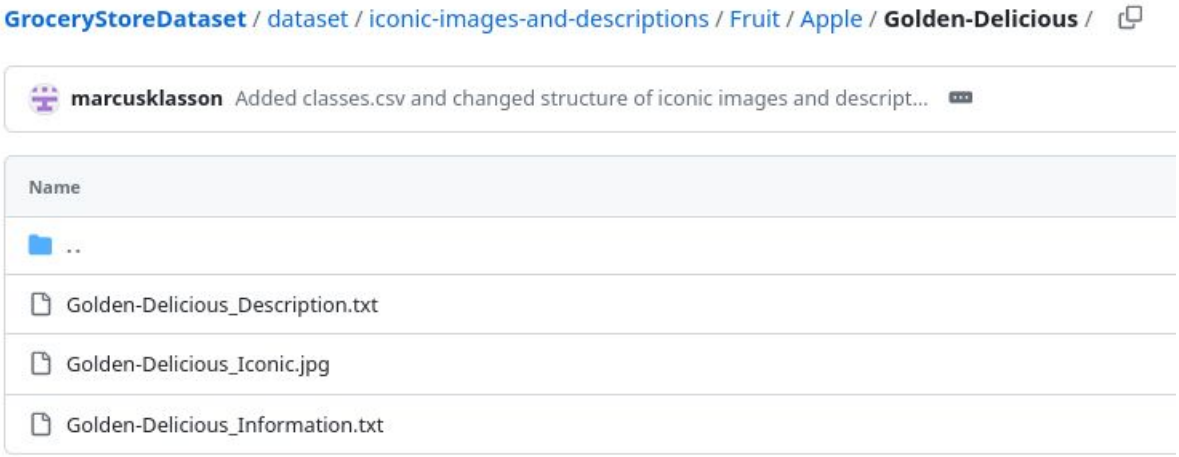


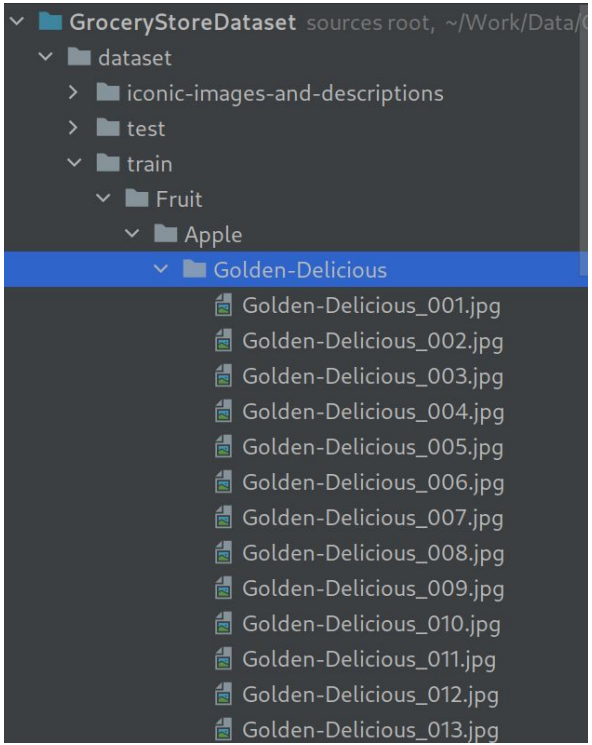
A multi-layer hierarchy of **natural**, **user-generated** images

By Marcus Klasson and Cheng Zhang and Hedvig Kjellström

Labeled data?

algolia
Powering discovery





YAGNI? No, YMANI!

algolia
Powering discovery

GroceryStoreDataset / dataset / iconic-images-and-descriptions / Fruit / Apple / Golden-Delicious / **Golden-Delicious_Description.txt**

marcusklasson Added classes.csv and changed structure of iconic images and descript...

Code Blame 1 lines (1 loc) · 147 Bytes Code 55% faster with GitHub Copilot

```
1  Golden Delicious has a white juicy pulp and a greenish yellow shell. The taste is mellow and sweet, making Golden Delicious suitable for desserts.
```

You Might Also Need It™

GroceryStoreDataset / dataset / iconic-images-and-descriptions / Fruit / Apple / Golden-Delicious / **Golden-Delicious_Information.txt**

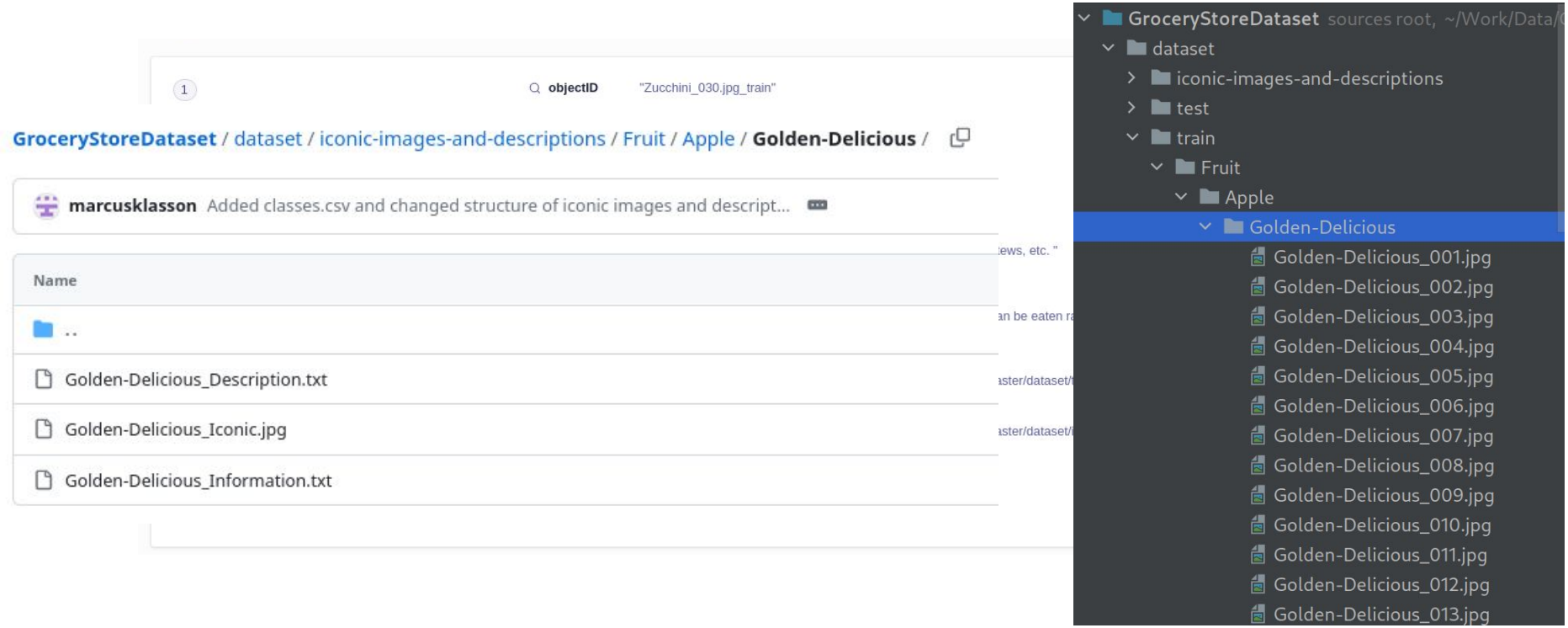
marcusklasson Added classes.csv and changed structure of iconic images and descript... bcd6345

Code Blame 4 lines (4 loc) · 315 Bytes Code 55% faster with GitHub Copilot Ra

```
1  Title: Apple Golden Delicious Class 1
2  Country and volume: Italy, ca 180g
3  Description: Golden Delicious has a white juicy pulp and a greenish yellow shell. The taste is mellow and sweet, making Golden Delicious suitable for desserts.
4  URL: https://www.hemkop.se/produkt/apple-Golden-Delicious-Klass-1-100142141_KG
```

Structuring data – a kind of Magic 🪄

algolia
Powering discovery



algolia
Powering Discovery

The million-dollar insight

Any **labeled** dataset is a **benchmark**
asking you to be **built**

algolia
Powering Discovery

“A **Benchmark** is only
a **dataset** with **testable labels**”

Paul-Louis Nech
ML Engineer

Step 1/2: structuring – index.py

algolia
Powering discovery

```python
def main():
    records = []
    datasets = ["test", "train", "val"]
    for dataset in datasets:
        records.extend(browse_dataset(dataset))
    index.save_objects(records)
    print(f"Done saving {len(records):04} {datasets} groceries.")
```

Step 1/2: structuring – index.py

algolia
Powering discovery

```python
def browse_dataset(dataset_name: Optional[str] = "train") -> list[dict[str, Any]]:
    records = []
    metadata = browse_metadata()
    datasets = "/home/pln/Work/Data/GroceryStoreDataset/dataset/"
    for dir, _, files in os.walk(os.path.join(datasets, dataset_name)):
        for i, file in enumerate(files, start=1):
            # Process filename
            # e.g. "Avocado_001.jpg" -> ("Avocado", "001")
            product, id = file.split("_")
            id = id.removesuffix(".jpg")

            # Process directory labels
            # e.g. train/Fruit/Apple/GrannySmith/ -> ("Fruit", "Apple", "GrannySmith")
            imgpath = dir.removeprefix("/home/pln/Work/Data/GroceryStoreDataset/")
            iconic_dir = imgpath.replace(f'dataset/{dataset_name}', 'dataset/iconic-images-and-
descriptions')
            iconic_path = f"{REPO_ROOT}/{iconic_dir}/{product}_Iconic.jpg?raw=true"

            # Edge case: some items have a category but no subCategory
            categories = dir.split("/")[-3:]
            if dataset_name in categories:
                categories = categories[1:]

            record = {...}

            records.append(record)

    records.sort(key=lambda x: x['objectID'])
    return records
```

Step 1/2: structuring – `index.py`

algolia
Powering discovery

1

objectID	"Zucchini_030.jpg_train"
image_id	"030"
category	"Vegetables"
sub_category	"Zucchini"
name	"Zucchini"
description	"Zucchini can be eaten raw, but is usually used in soups, pots, stews, etc. "
sku	"Zucchini"
information	{ title: "Zucchini Class 1", country: "Spain", description: "Zucchini can be eaten raw, but is usually used in soups, pots, stew
dataset	"train"
image	"https://github.com/marcusklasson/GroceryStoreDataset/blob/master/dataset/train/Vegetables/Zucchini/Zucchini_030.jpg?raw=true"
iconic	"https://github.com/marcusklasson/GroceryStoreDataset/blob/master/dataset/iconic-images-and-descriptions/Vegetables/Zucchini/Zucchini_Iconic.jpg?raw=true"

You Might Also Need It™

Show fewer attributes

algolia
Powering Discovery

We can now evaluate

But *what* do we want to measure?

What do we want to know about our algorithm?

algolia
Powering discovery

We want to assess quality of recommendations on groceries

- ➔ That is, we care about **precision** of categories (🍎 -> only 🍎🍏🍎, no 🌭)
- ➔ Not that interested in their **recall** (🍎 -> **only *apples*** >> 🍎 -> **all** the ***apples***)
- ➔ Customers also care about **always showing something** (📉 no results rate)

➔ Our metrics:

`GotResults`, `SameCategory`,
`SameSubCategory`, `SameProduct`

Step 2/2: evaluating – bench.py

algolia
Powering discovery

```python
for item in dataset:
    query = { ... }
    response = client.get_recommendations([query])["results"][0]
    recos = response["hits"]
    if recos: # Let's check the quality of recommended items!
        # ...
    else:
        # Let's report failed items for further investigation.
        no_match.append(item)

metrics.compute(no_match, same_cat, same_sub, same_sku)

print(
    f"Benchmark@{at_k} results - {metrics.nb_items} items\n"
    f"- Got some results {metrics.ratio_predicted:.2%} of the time.\n"
    f"- Predicted same category {metrics.ratio_same_category:.2%} of the time.\n"
    f"- Predicted same sub-cat  {metrics.ratio_same_subcategory:.2%} of the time.\n"
    f"- Predicted same product  {metrics.ratio_same_sku:.2%} of the time.\n"
)
```

Step 2/2: evaluating – bench.py

algolia
Powering discovery

```python
for reco_item in recos:
    reco_item_cat = reco_item["category"]
    reco_item_sub = reco_item["sub_category"]
    reco_item_sku = reco_item["sku"]
    is_same_cat = item_cat == reco_item_cat
    is_same_sub = item_sub == reco_item_sub
    is_same_sku = item_sku == reco_item_sku
    same_cat.append(is_same_cat)
    same_sub.append(is_same_sub)
    same_sku.append(is_same_sku)

    if not is_same_cat:
        if is_same_sub:
            print("WTF??")
            wrong_cat.append((item, reco_item))
    if not is_same_sub:
        if is_same_sku:
            print("WTF??")
            wrong_sub.append((item, reco_item))
        if not is_same_sku:
            wrong_sku.append((item, reco_item))
```

GARBAGE IN,
GARBAGE OUT

A small tip: make it reliable

algolia
Powering discovery

Once your benchmark runs, make it **reliable**

same model, same input, same metric
should result in **same scores**

-> Freeze your GPT4 **model version**
-> use `random.seed(42)`
-> etc - eliminate as much **randomness** as you can

Step 2/2: evaluating – bench.py demo!

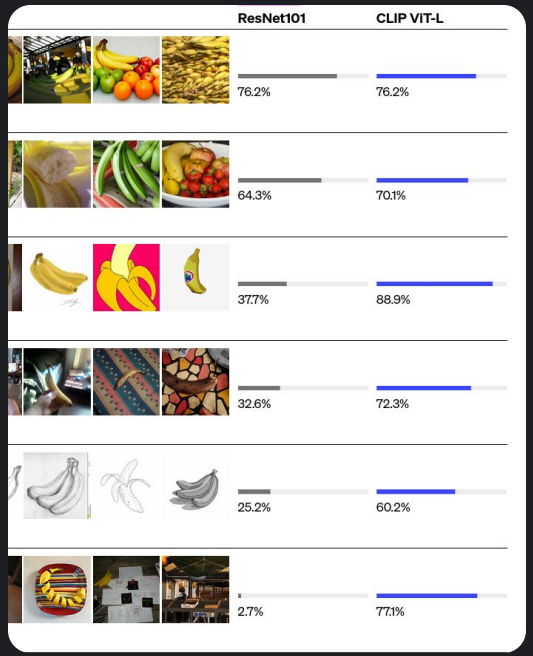
algolia
Powering discovery

```python
if __name__ == "__main__":
    at_k = [1, 3, 5, 10, 30]

    bench_predictions( filename: "predictions/CLIP_7510047928944698635.json", at_k=at_k)
    bench_predictions( filename: "predictions/ResNet_7510077461642073363.json", at_k=at_k)

```

Step 2/2: evaluating – bench.py demo!

algolia
Powering discovery

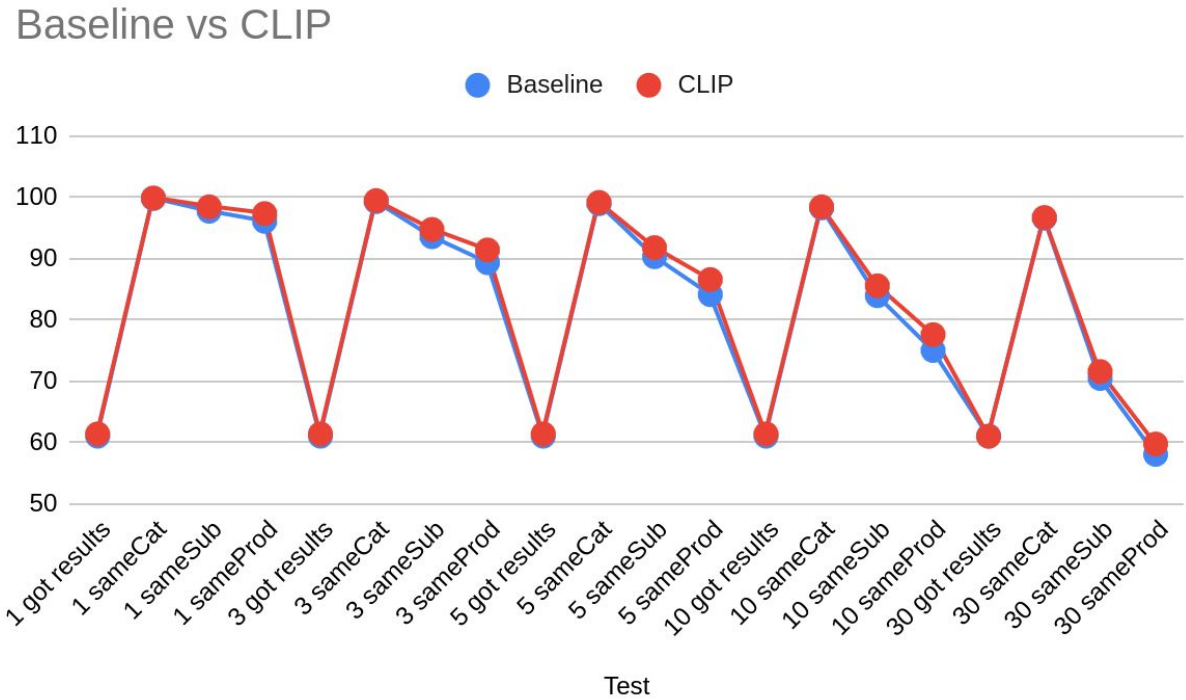
```
Begin of benchmark for predictions in file predictions/ResNet_7510077461642073363.json.
Fetching up to 30 recos for 5420 items...
Starting benchmark@1 on index `groceries` over 5420 items.
Benchmark@1 results - 5420 items
- Got some results 99.28% of the time.
- Predicted same category 99.63% of the time.
- Predicted same sub-cat  96.04% of the time.
- Predicted same product  92.83% of the time.

Starting benchmark@3 on index `groceries` over 5420 items.
Benchmark@3 results - 5420 items
- Got some results 99.28% of the time.
- Predicted same category 98.98% of the time.
- Predicted same sub-cat  90.81% of the time.
- Predicted same product  84.28% of the time.
```



Step 2/2: evaluating – bench.py results

algolia
Powering discovery

Test	Baseline	CLIP



30 sameProd	58.1	59.8

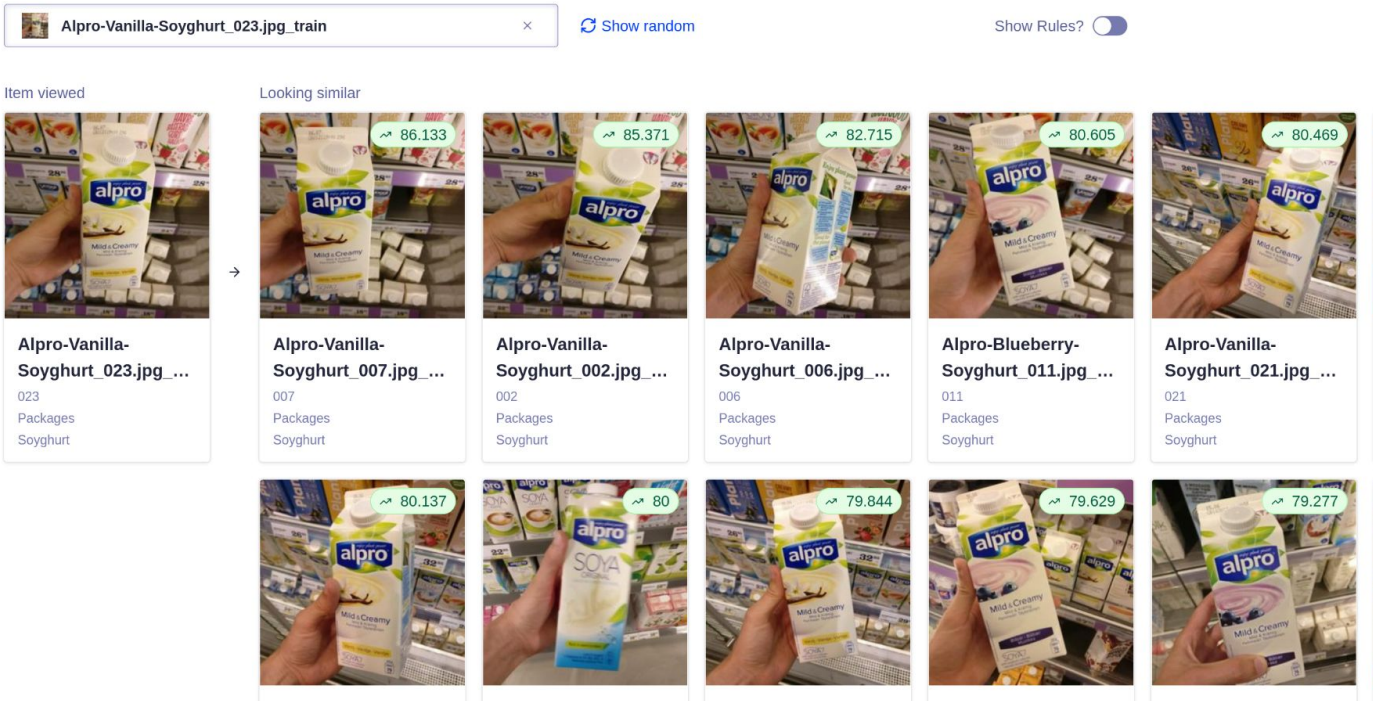
Non-**significant** difference

And **lower stability** (when multi threading)

algolia
Powering Discovery

Some evaluation results

Evaluation results: packaged goods?



Reality: packaged goods are hard to recognize

Expectation: packaged goods are hard to recognize

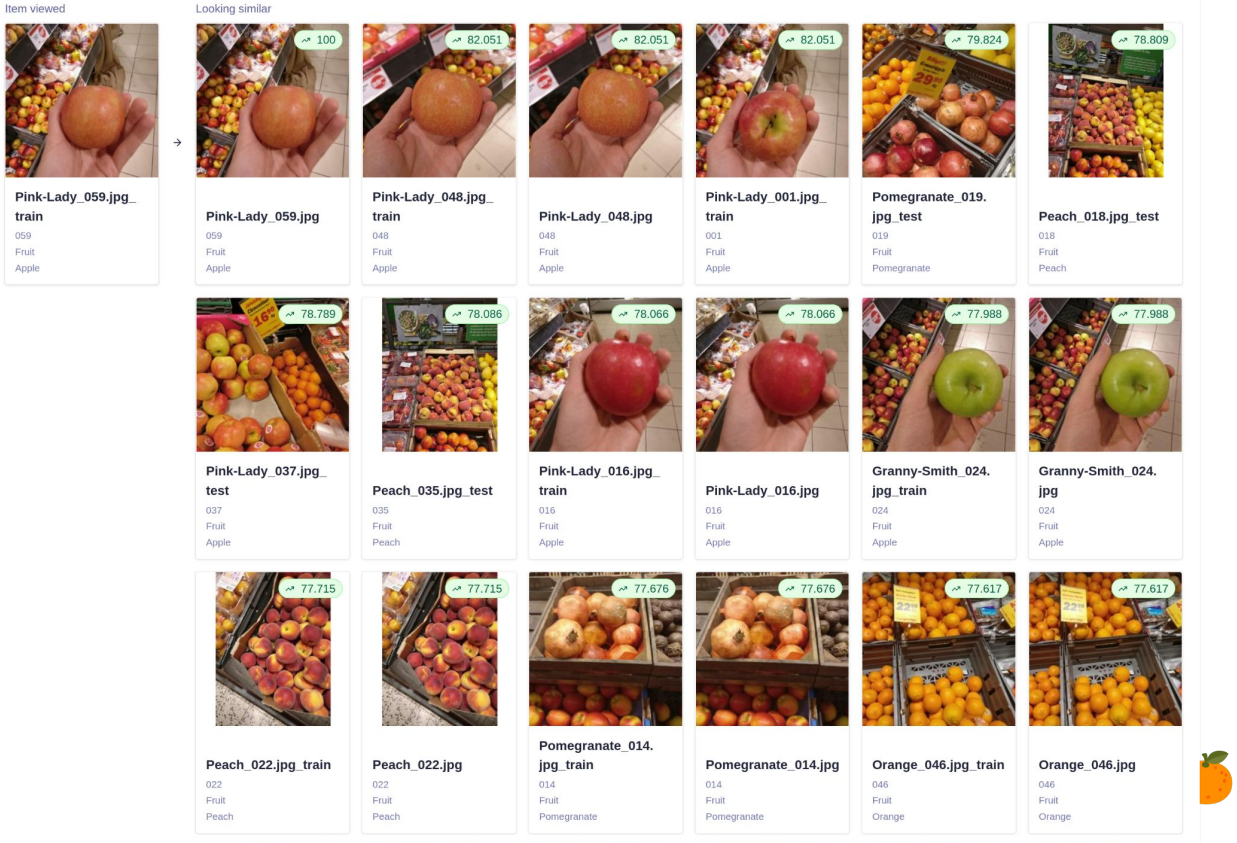
Evaluation aside: TVs vs Ovens

algolia
Powering discovery

⁉️

Evaluation results: comparing 🍎 to 🍊?

algolia
Powering discovery



algolia
Powering Discovery

"An **AI model**'s **strengths** and **weaknesses** are not what *we, humans,* think they are."

Paul-Louis Nech
ML Engineer

algolia
Powering Discovery

Communicating the results

Optimize for action

algolia
Powering Discovery

"We measured
CategoryPrecision@1 = 99.62 🎉"

An engineer
Pretty proud of their metrics

"The **top recommendation** is in the **same category** as the source item for **99.62%** of **all pictures**"

A *senior* engineer
Who knows how to share metrics

"**WTF**? Can I speak to a *human* on your team?"

A customer
Pretty confused right now

"Thanks! I'll **trust your API** with my precious use-case 🥒🍎🍏"

A *happy* customer
Confident in your feature

algolia
Powering Discovery

“**WTF**?Am I talking to a bad LLM?”

A customer
Pretty confused right now

algolia
Powering Discovery

"The **top recommendation** is in the **same category** as the source item for **99.62%** of **all pictures**"

A senior engineer
Who knows how to share metrics

algolia
Powering Discovery

“Thanks! I can trust it with my precious use-case 🌭”

A happy customer
Confident in your feature

Your interlocutors *don’t want metrics*

algolia
Powering discovery

They want a *quantitative assessment*
of the **usefulness of your product** 💡

➔ Focus on making your report **actionable**

What do you expect readers to **do** once they finish reading?

How can you **optimize for action**? ✊

What do we want to say about our algorithm?

algolia
Powering discovery

Communicate a metric tied to customer value

- ➔ **Precision@1** is a very **good-looking** metric, also pretty **useless**
- ➔ **Precision@3** is the customers' **main use-case**: carousel with few top hits
- ➔ **Precision@30** is the **hard truth**: up to maxRecs, how many are good?

➔ It depends **what** you want to communicate and **to whom**

To whom?

algolia
Powering discovery

"SameProduct@3 is **88.64**%, let's try **XYZ** and check if precision **goes up**"

To my dev colleagues

"In our **Carousel UI components**, we can expect **80% of top 3 results** to be the same SKU as user input"

To our Product Manager

"You can be **confident** that on **most groceries**, your users will see **relevant recommendations**"

To our Customer

Who are you talking to? Interlocutor Personas

algolia
Powering discovery

Technical audiences

Want to know how to **run it**
Would love to **adapt** it to new features
Curious what can be **improved**

- ➔ Share **hands-on** knowledge 🛠️
- ➔ Make the code **modular** 🧱
- ➔ Mention every **limitation** ⚠️

Non-Technical audiences

Want to know how to **leverage it**
Need **actionable talking points**
(especially customer-facing roles)
Must know **core strengths**

- ➔ Make the report **self-serve**
- ➔ Give them **ammunition**
- ➔ Cover use-cases where **feature shines** and those **best avoided** for now

Who are you talking to? Internal/External

algolia
Powering discovery

Internal audiences

Needs to see strengths and weaknesses in **crude light**, to know what needs **focus**

- ➔ Be **exhaustive** in what you report
- ➔ Focus on the **edge-cases**
- ➔ Share ideas for **improving these** weak spots of your feature

External audiences

Needs to know **where** it can trust you **today**
Focus on the **solid use-cases**

- ➔ Ace **core-use cases** with **great docs**
- ➔ When possible, explain **workarounds**
- ➔ Otherwise, mention **alternative features** better suited to this case

algolia
Powering Discovery

Iterating on our model

Now we have a Baseline

First, the good news: we do well on food 🍎🥒🌭

algolia
Powering discovery

Item viewed

Zucchini_027.jpg_
test
027
Vegetables
Zucchini

Looking similar

Zucchini_021.jpg_
test
021
Vegetables
Zucchini

Zucchini_020.jpg_
test
020
Vegetables
Zucchini

Zucchini_025.jpg_
test
025
Vegetables
Zucchini

Zucchini_006.jpg_
test
006
Vegetables
Zucchini

Zucchini_010.jpg_
train
010
Vegetables
Zucchini

Zucchini_009.jpg_
train
009
Vegetables
Zucchini

Zucchini_014.jpg_
train
014
Vegetables
Zucchini

Zucchini_028.jpg_
test
028
Vegetables
Zucchini

Zucchini_001.jpg_
train
001
Vegetables
Zucchini

Zucchini_001.jpg_
test
001
Vegetables
Zucchini

Zucchini_017.jpg_
train
017
Vegetables
Zucchini

Zucchini_012.jpg_
train
012
Vegetables
Zucchini

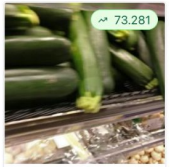
Zucchini_016.jpg_
train
016
Vegetables
Zucchini

Zucchini_017.jpg_
test
017
Vegetables
Zucchini

Zucchini_020.jpg_
train
020
Vegetables
Zucchini

Zucchini_007.jpg_
train
007
Vegetables
Zucchini

Zucchini_018.jpg_
train
018
Vegetables
Zucchini



Zucchini_001.jpg_val
001
Vegetables
Zucchini

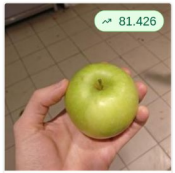
First, the good news: we do well on food 🍎🥒🌭

algolia
Powering discovery

Item viewed

Granny-Smith_059.jpg_train
059
Fruit
Apple

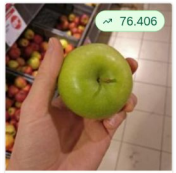
Looking similar



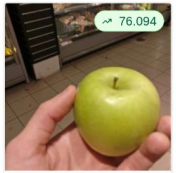
Granny-Smith_017.jpg_train
017
Fruit
Apple

Granny-Smith_015.jpg_train
015
Fruit
Apple

Granny-Smith_033.jpg_train
033
Fruit
Apple



Granny-Smith_050.jpg_train
050
Fruit
Apple

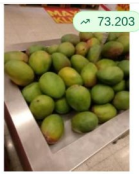


Granny-Smith_040.jpg_train
040
Fruit
Apple

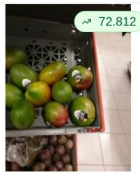
Golden-Delicious_028.jpg_train
028
Fruit
Apple

Granny-Smith_021.jpg_train
021
Fruit
Apple

Golden-Delicious_003.jpg_train
003
Fruit
Apple



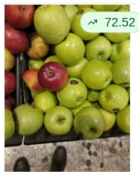
Mango_017.jpg_test
017
Fruit
Mango



Mango_020.jpg_test
020
Fruit
Mango

Golden-Delicious_003.jpg_val
003
Fruit
Apple

Granny-Smith_024.jpg_train
024
Fruit
Apple



Granny-Smith_001.jpg_test
001
Fruit
Apple

Papaya_006.jpg_test
006
Fruit
Papaya

Golden-Delicious_004.jpg_val
004
Fruit
Apple

Granny-Smith_003.jpg_val
003
Fruit
Apple

Golden-Delicious_014.jpg_train
014
Fruit
Apple

Granny-Smith_023.jpg_test
023
Fruit
Apple

Some edge-cases where the model is confused

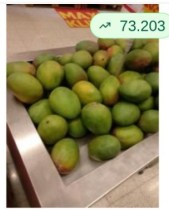
algolia
Powering discovery

Item viewed

Granny-Smith
Granny-Smith_059.jpg_train

→

Golden-Delicious
Golden-Delicious_028.jpg_tr…



Mango
Mango_017.jpg_test

Papaya
Papaya_006.jpg_test

→ GotResults ✅
SameCategory ✅ (3/3)
SameSubCategory ✅ (2/3)
SameProduct ❌ (1/3)

How can we solve these?

Item viewed

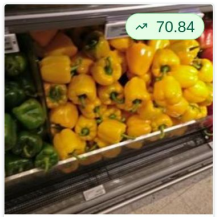
Zucchini_030.jpg_train
030
Vegetables
Zucchini

→

Looking similar

Green-Bell-Pepper_022.jpg_test
022
Vegetables
Pepper

Zucchini_022.jpg_train
022
Vegetables
Zucchini

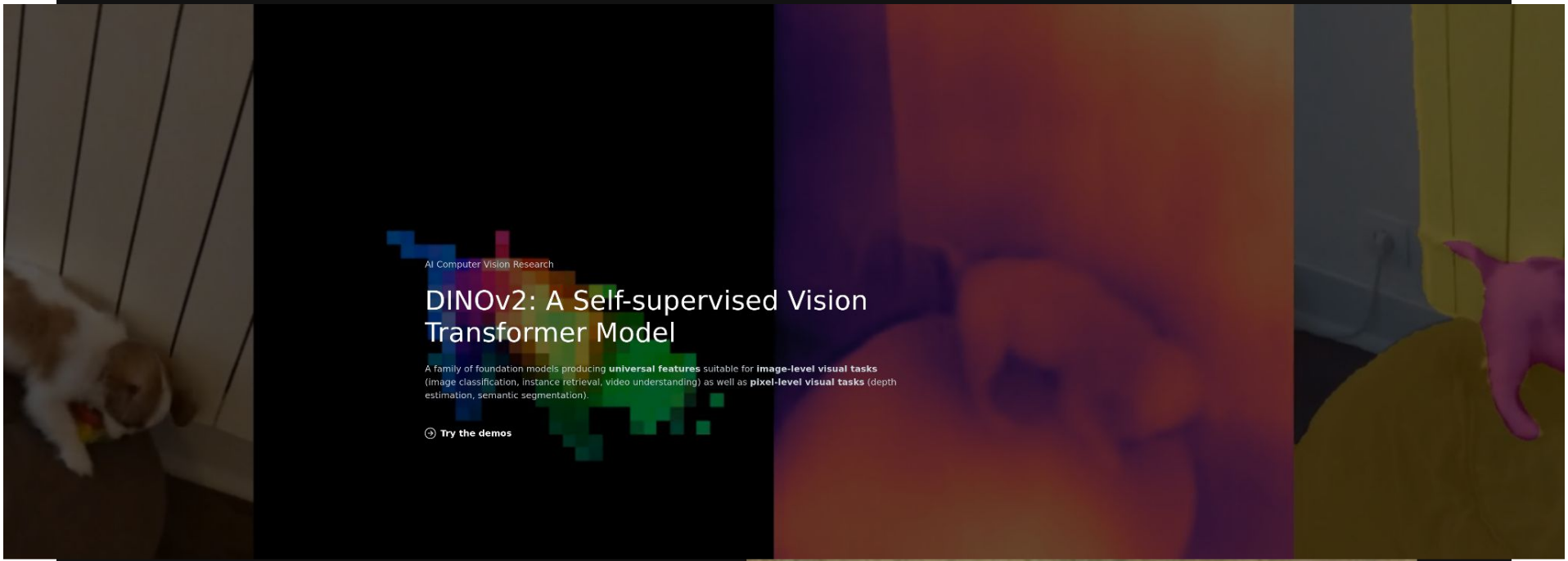


Yellow-Bell-Pepper_022.jpg_test
022
Vegetables
Pepper

→ GotResults ✅
SameCategory ✅ (3/3)
SameSubCategory ❌ (1/3)
SameProduct ❌ (1/3)

Addressing edge-cases: different embedding?

algolia
Powering discovery



Addressing edge-cases: different embeddings?

algolia
Powering discovery

Should we solve these edge-cases with **different embeddings**?

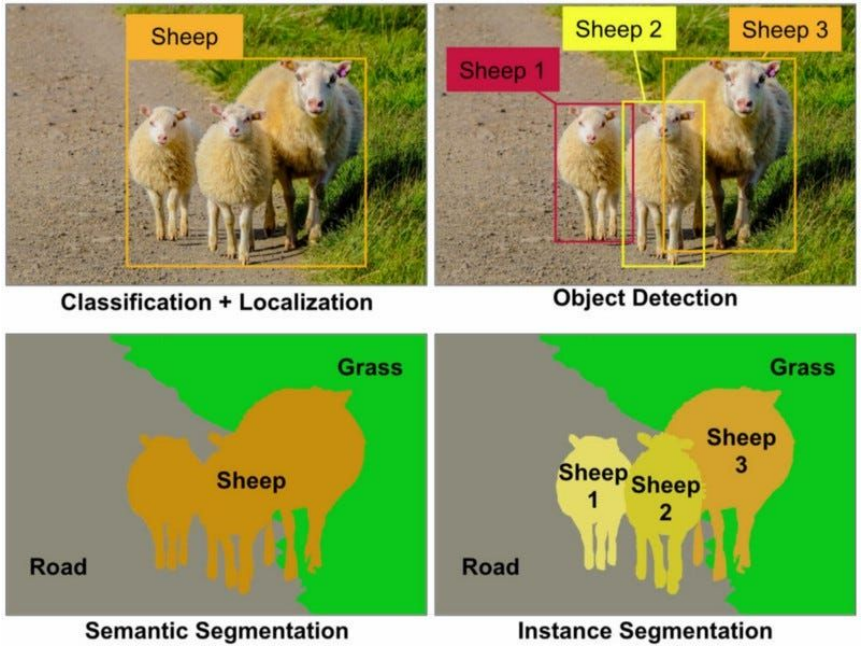
A **big move** with likely **side effects**

What else can we try first?

- **Image Preprocessing**
- **Object Segmentation**
- **Adding textual signal in vectors**

Addressing edge-cases: object segmentation?

algolia
Powering discovery



Can object segmentation help do a better job?

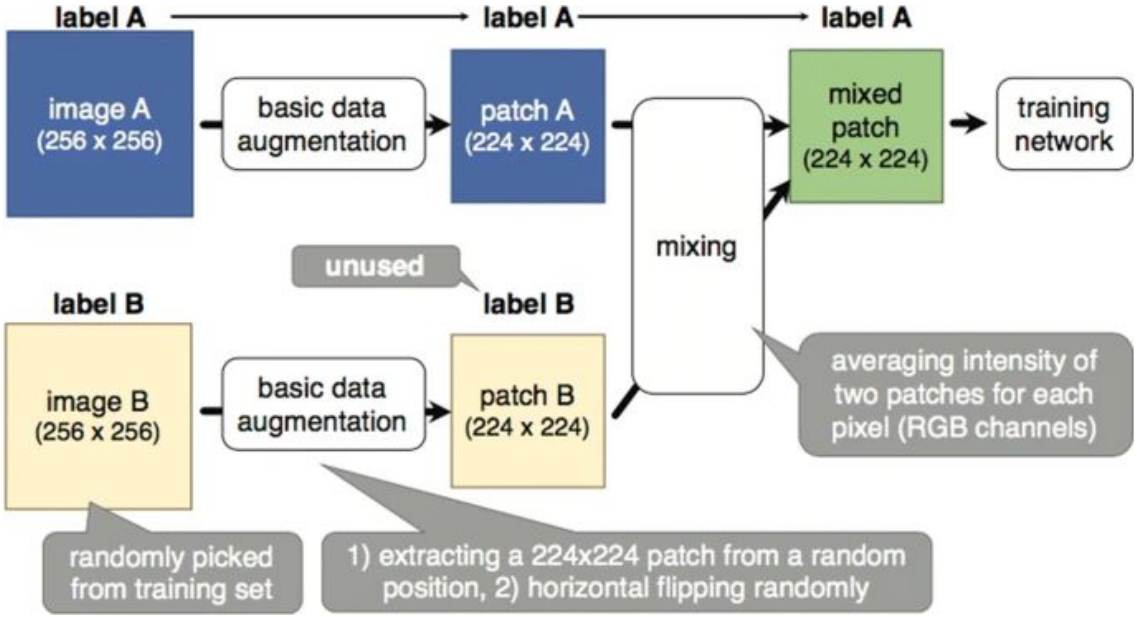
We can try various approaches to segmentation as a **preprocessing** step, and given the **same vector embedding model** see if it improves **precision**

Addressing edge-cases: data transformation?

algolia
Powering discovery

We can try data transformation techniques, from **basic image manipulation** (contrast, saturation, etc)

to **more opinionated ones** (e.g. when a record has 3 images, today we index separately and return the overall best score - how about **MixingImages**?



algolia
Powering Discovery

We can get creative

You can afford to experiment

algolia
Powering Discovery

You can do it too

What customer problems do you solve?

algolia
Powering discovery

- Try to find data that describes **good/bad outcomes** 📝
  - Else, search for data close to the **use-case you're trying to solve** 🧭

- What does **success** *look like*? *This* is your **metric**!
  - **"always having something to show"** -> `NoResultsRate@K` 📉
  - **"never showing Apple to Samsung products?"** -> `CategorySpread@K` 📉
  - **"optimize for TopCategoryXXX while keeping average quality under control?"**
    -> **two metrics** `TopCategoryXXXRecall` 📈 while keeping `CategoryPrecision` 🔼

A metric is **right** when it captures the **value you build**

Conclusion: You *can* build a benchmark

Remember: a **hacky** benchmark is better than **nothing**

hence **any data is better than no data**!

algolia

Thank you

Grab my slides

💌 pln@algolia.com

𝕏 @PaulLouisNech

Try our model