



AND OTHER THINGS

Counting Koalas with Kafka

 @SimonAubury
 @saisbury

Apache Kafka Meetup -
December 2023



Simple
Machines

Simon Aubury

Director of data platforms

♥ Kafka enthusiast

🤗 Confluent Community Catalyst

🏡 Sydney, Australia



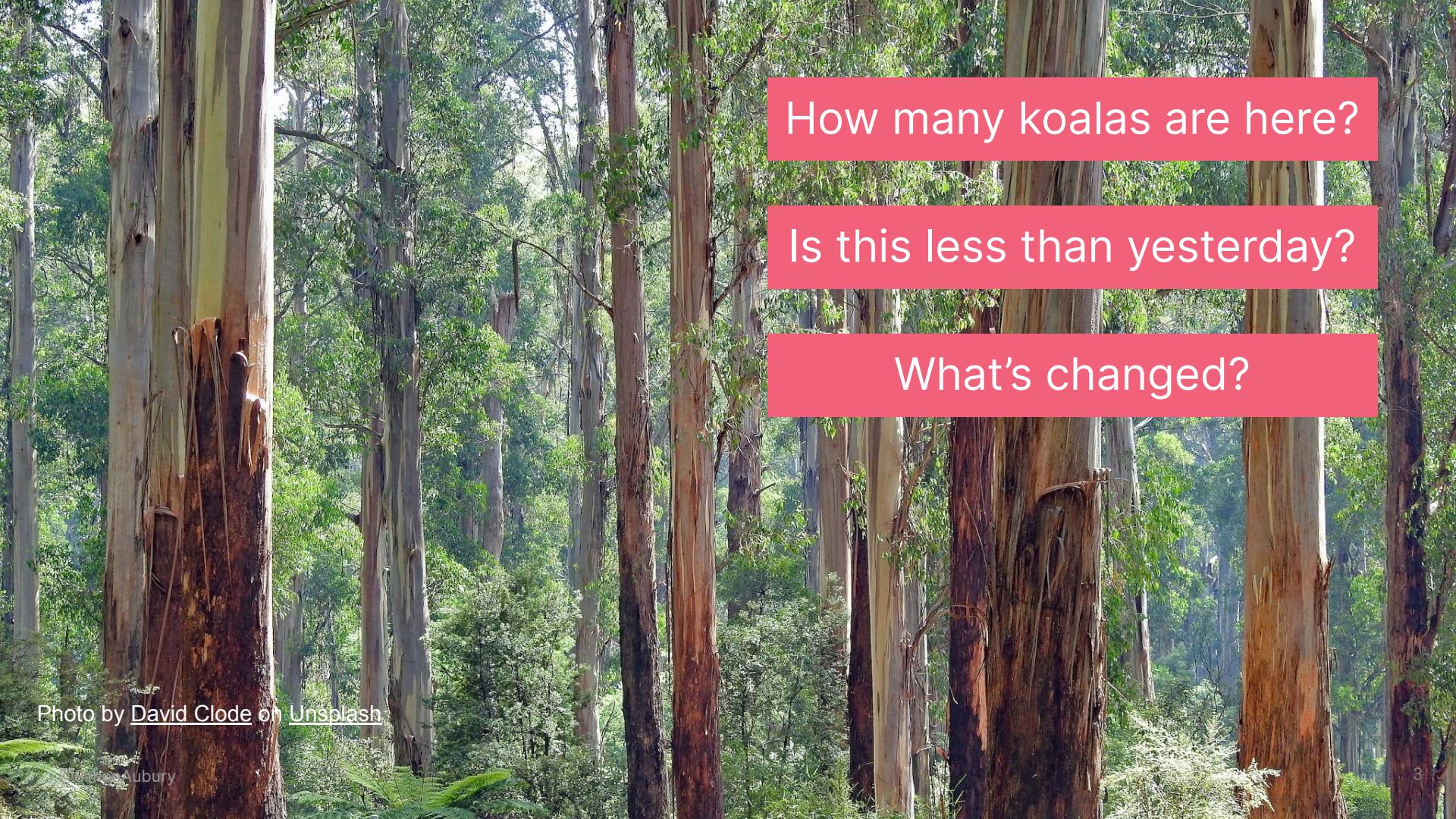
Baz

Southern koala

♥ Endangered

🤗 Eucalyptus trees

🏡 Australia

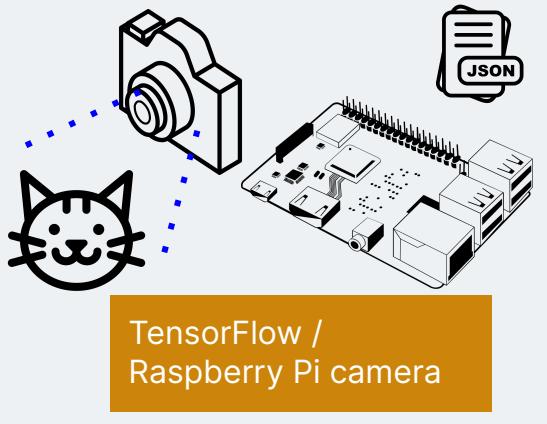
A photograph of a dense forest of tall eucalyptus trees. The trees have characteristic white, papery bark that is peeling in places, revealing a darker, reddish-brown wood underneath. The forest floor is covered with green ferns and smaller plants. The sunlight filters through the canopy of leaves at the top of the trees.

How many koalas are here?

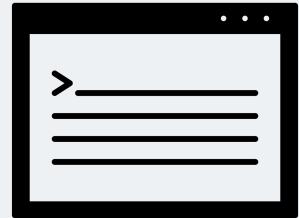
Is this less than yesterday?

What's changed?

Photo by [David Clode](#) on [Unsplash](#)



Console



1. Kafka-Cat



Raspberry Pi

Cameras in the wilderness

Raspberry Pi is a **low cost** credit-card sized computer

With an attached camera they have sufficient processing power for edge ML detection with TensorFlow Lite



TensorFlow

TensorFlow Lite object detection

Identify which of a known set of objects might be present and provide information about their positions within the image.

```
{ "cat": 0.82 }
```

```
{"cat":1, "koala":0}
```



Open Camera

```
# Start capturing video input from the camera  
cap = cv2.VideoCapture()
```

Load model

```
# Load model efficientdet_lite0.tflite, set threshold  
detector = vision.ObjectDetector.create_from_options(70%)
```

Single image

```
# Continuously capture camera images  
while True:  
    image = cap.read()
```

Object detect

```
# Run object detection estimation using the model  
detection_result = detector.detect(image)
```

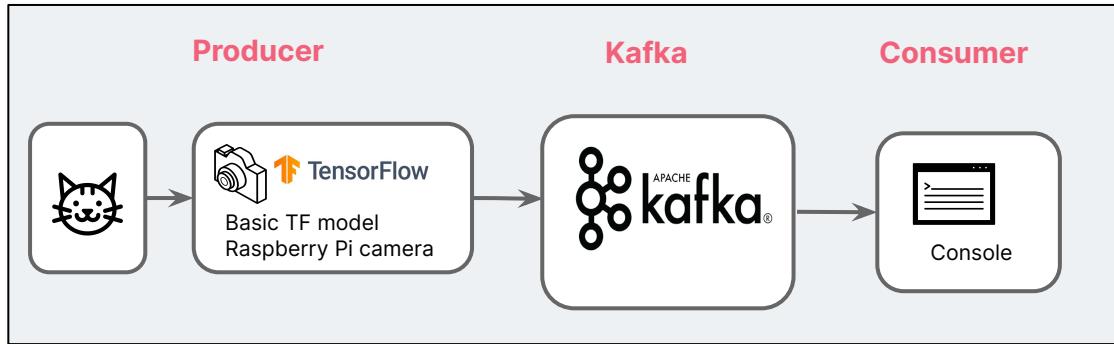
Kafka produce

```
# Produce to objects topic  
kafka_producer.produce( detection_result )
```

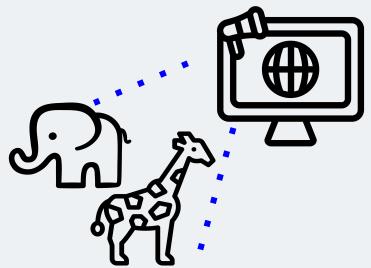
<https://github.com/saubury/wildlife-watch/blob/main/detect.py>

Architecture

MVP version



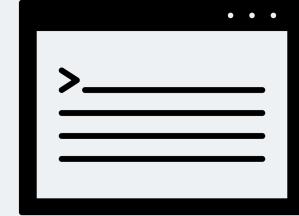
```
-zsh
Python          #1           -zsh
saubury:wildlife-watch % kafka-console-consumer --bootstrap-server localhost:9092 --topic objects
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.74}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.74}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.74}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.76}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.76}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.76}], "objects_count": {"cat": 1}}
{"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.72}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.76}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.72}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.72}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.71}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.71}], "objects_count": {"cat": 1}}
 {"camera_name": "raspberry-pi", "objects_found": [{"class_name": "cat", "probability": 0.71}], "objects_count": {"cat": 1}}
```



TensorFlow / Zoo
Livestream



Console



2. Zoo-keeping with ksqlDB

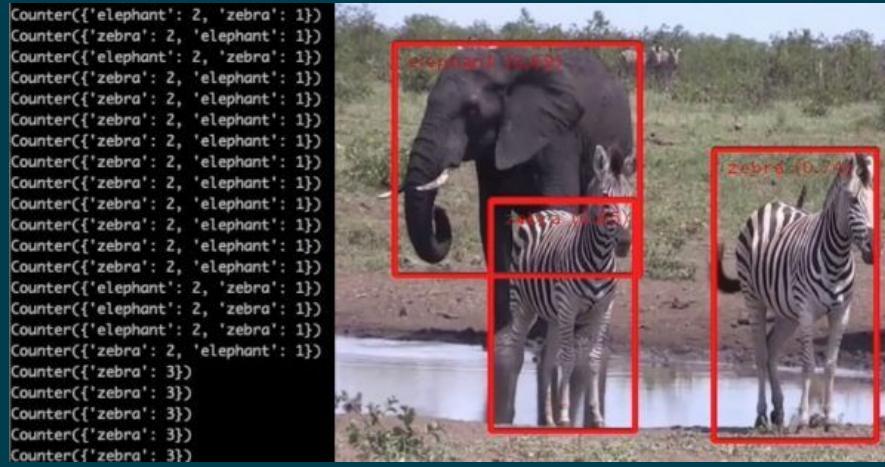


Zoo camera feed

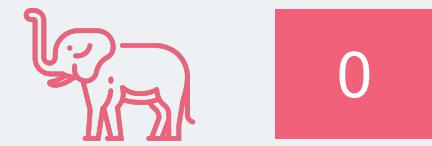
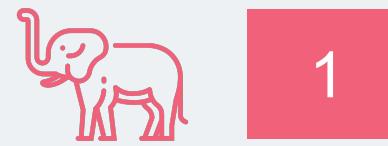
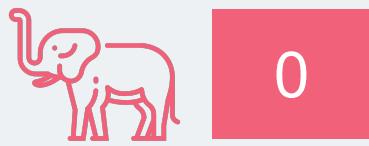
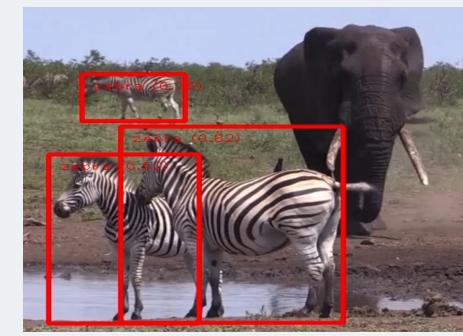
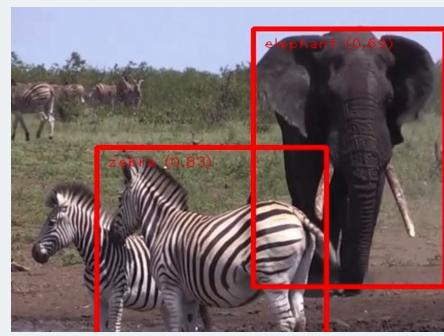
Expanding the animal collection

More animals from the zoo

- Additional Kafka producer
- Laptop based
- Source video webcam feed from local zoo ([Pyautogui & OpenCV](#))



```
{  
    "camera_name": "zoo-webcam",  
    "objects_count": {  
        "elephant": 1,  
        "zebra": 2  
    }  
}
```



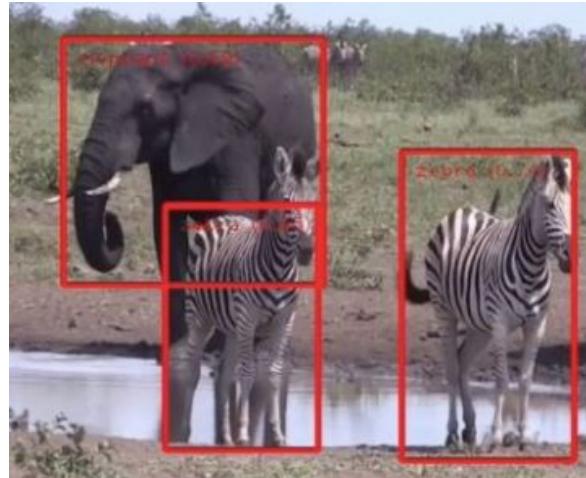
Payload extraction

Payload

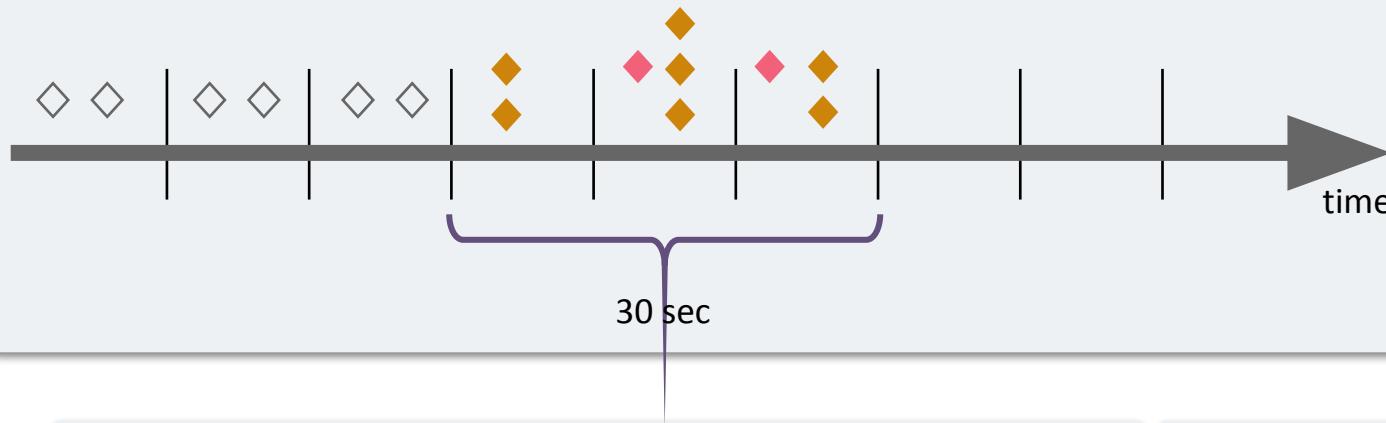
```
{  
    "camera_name": "zoo-webcam",  
    "objects_count": {  
        "elephant": 1,  
        "zebra": 2  
    }  
}
```

KSQL pivot

```
create stream animals as  
select extractjsonfield(objects_count, '$.elephant') as elephant  
, extractjsonfield(objects_count, '$.zebra') as zebra  
, < many more animals >  
from objects;
```



ksqlDB



KSQL table

```
create table zoo as
select max(elephant), max(zebra)
from animals window tumbling (size 30 seconds)
group by camera_name
emit changes;
```



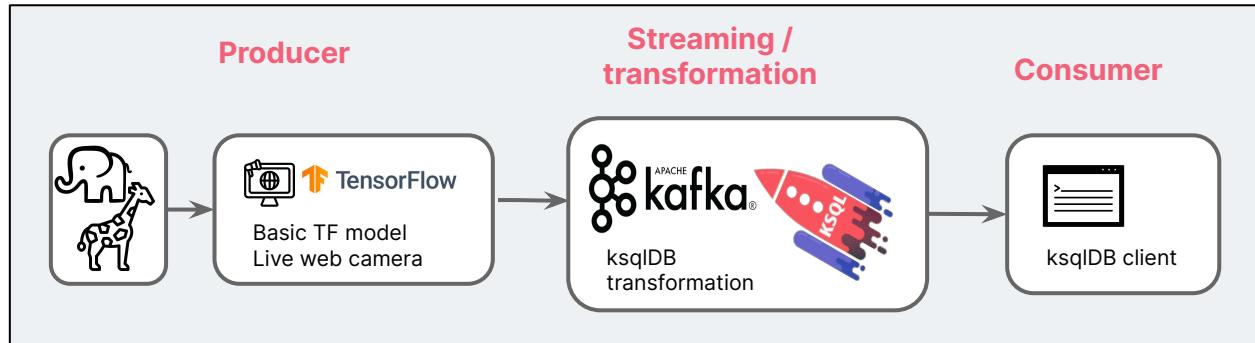
1



3

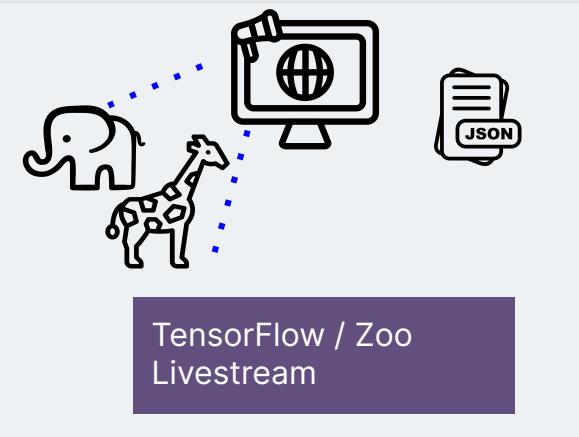
Architecture

Zoo-keeper version



```
ksql> select camera_name, max_elephant, max_zebra from zoo emit changes;
```

CAMERA_NAME	MAX_ELEPHANT	MAX_ZEBRA
mylaptop	null	1
mylaptop	null	2
mylaptop	null	3
mylaptop	1	3
mylaptop	1	3
mylaptop	1	3



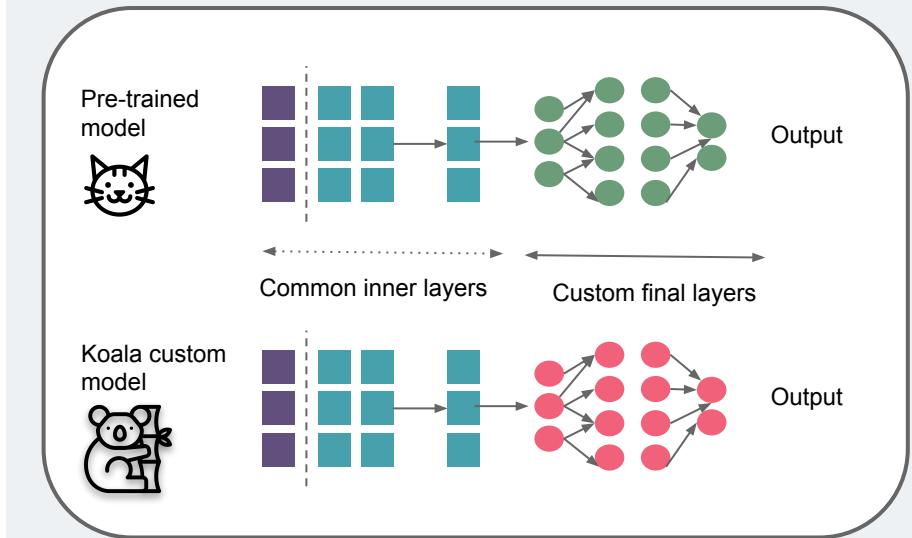
3. Transfer learning with Koalas



Transfer learning

Custom Koala object detection model

i Transfer learning is an ML technique that focuses on using knowledge gained while solving one problem ... and applying it to a different but related problem.

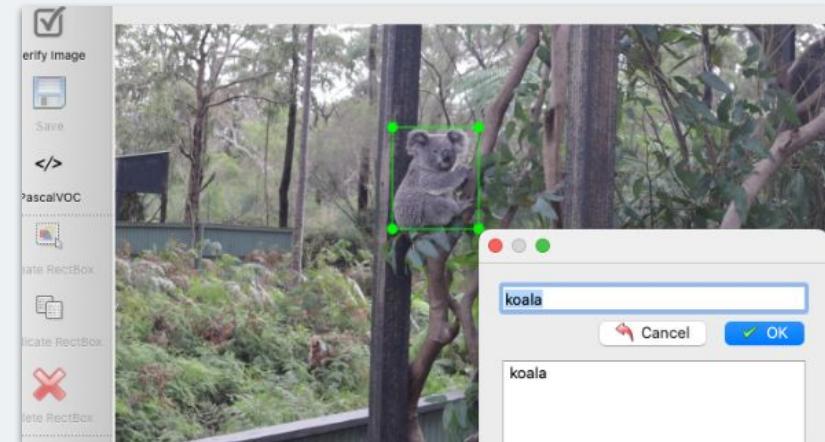


Transfer learning

Custom Koala object detection model

▲ EfficientDet TensorFlow Lite trained on COCO 2017 dataset ... over 200K labeled images

🎯 Goal - retrain an model with koala dataset to train a custom object detection model



Edges, shapes,
outlines,
contrasts



Koala specific
data set

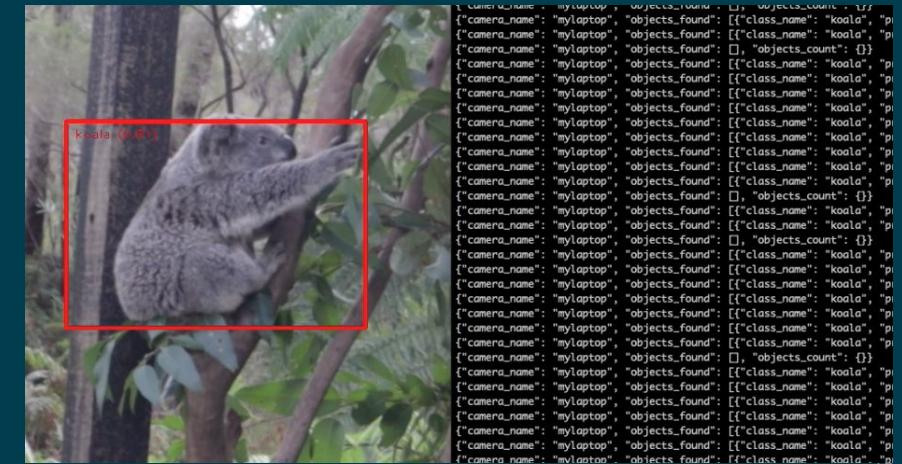
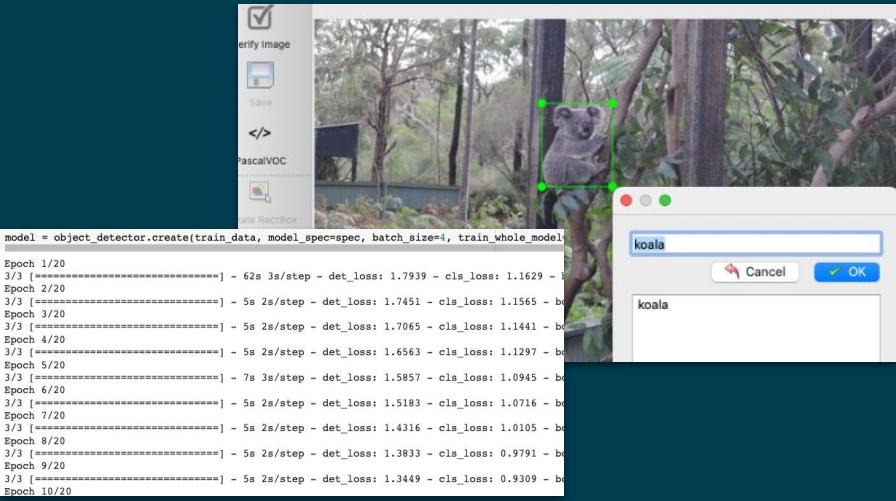
Koala model

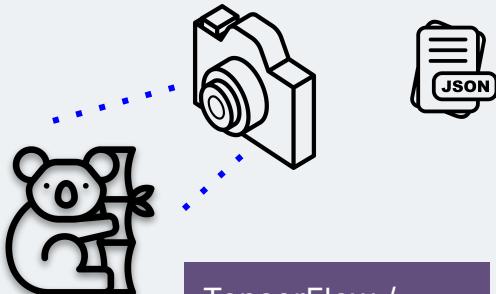
Retraining a TensorFlow Lite model



TensorFlow Lite Model Maker

- Koala dataset with [LabellImg](#)
 - Train the TensorFlow model
 - Export as a TensorFlow Lite model.
 - Evaluate model
 - Deploy model to RaspberryPi





Kafka connect / elastic sink



Kafka connect / http sink



4. Analysis and alerting

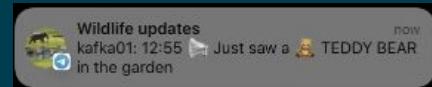
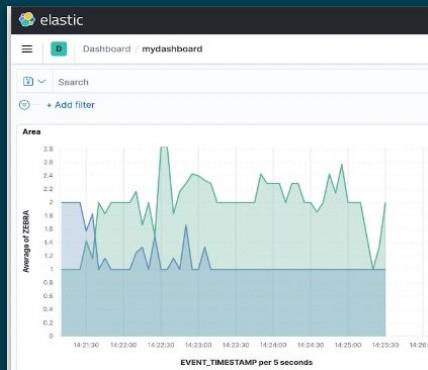


Analysis and alerting

Command line is cool, but ...

- Real-time dashboard with [Kibana](#)
- Phone alerts with [Telegram](#)

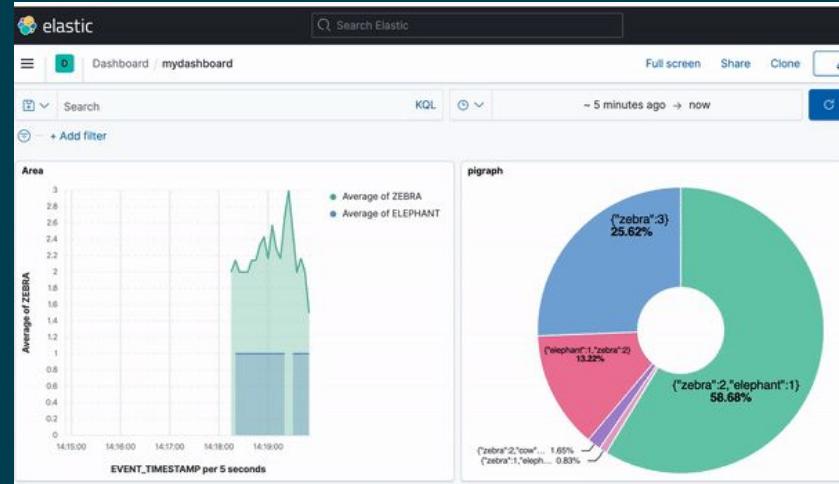
[Kafka Connect](#) is a framework for connecting Kafka with external systems



Kibana dashboard



Kafka Connect with Kafka Connect
Elasticsearch connector to send both
the **animals** and **zoo** Kafka topics to
Elasticsearch indexes.



Building the dashboard

Kafka connect / elastic sink

Kafka Topic

```
{  
  "topics": "ZOO",  
  "key.converter": "org.apache.kafka.connect.storage.StringConverter",  
  "value.converter.schemas.enable": "false",  
  "connector.class": "io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",  
  "key.ignore": "true",  
  "value.converter": "org.apache.kafka.connect.json.JsonConverter",  
  "type.name": "type.name=kafkaconnect",  
  "topic.index.map": "ZOO:zoo",  
  "connection.url": "http://elasticsearch:9200",  
  "transforms": "ExtractTimestamp",  
  "transforms.ExtractTimestamp.type": "org.apache.kafka.connect.transforms.InsertField$Value",  
  "transforms.ExtractTimestamp.timestamp.field": "EVENT_TIMESTAMP"
```

Elastic sink class

Index name

Timestamp

ZOO*

Time field: EVENT_TIMESTAMP

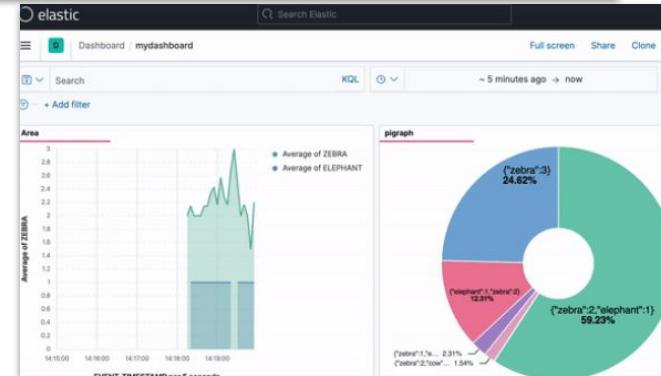
This page lists every field in the **zoo*** index and the field's associated core type as recorded by Elasticsearch Mapping API

Fields (8) Scripted fields (0) Field filters (0)

Search

Name	Type	Format	Searchable
EVENT_TIMESTAMP	date		●
MAX_ELEPHANT	number		●
MAX_ZEBRA	number		●

@SimonAubury



Telegram bot



Created wildlife Telegram bot with exposed **HTTP-based** interface

Telegram bot alert for each record in **teddybear-telegram-topic** Kafka

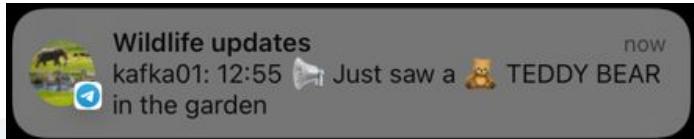


Teddy bear alerts

Kafka connect / http sink

KSQL stream

```
create stream teadybear-telegram-topic
as
select 'Just saw a 🐻 TEDDY BEAR in the garden' as message
from animals
where teddybear > 0 ;
```



Kafka topic

```
{
  "topics": "teadybear-telegram-topic",
  "input.data.format": "JSON",
  "connector.class": "HttpSink",
  "confluent.topic.bootstrap.servers": "broker:29092",
  "confluent.topic.replication.factor": "1",
  "reporter.bootstrap.servers": "broker:29092",
  "reporter.result.topic.name": "success-responses",
  "reporter.result.topic.replication.factor": "1",
  "reporter.error.topic.name": "error-responses",
  "reporter.error.topic.replication.factor": "1",
  "http.api.url": "https://api.telegram.org/botXXXXXX/sendMessage",
  "request.method": "POST",
  "headers": "Content-Type: application/json",
  "request.body.format": "string",
  "batch.max.size": "1",
  "batch.prefix": "{\"chat_id\": \"-123456\", ",
  "batch.suffix": "}",
  "regex.patterns": ".+MESSAGE=(.+)",
  "regex.replacements": "\"text\": \"$1\"",
  "regex.separator": "~",
  "tasks.max": "1",
  "value.converter": "org.apache.kafka.connect.storage.StringConverter",
  "name": "teadybear-telegram-sink"
}
```

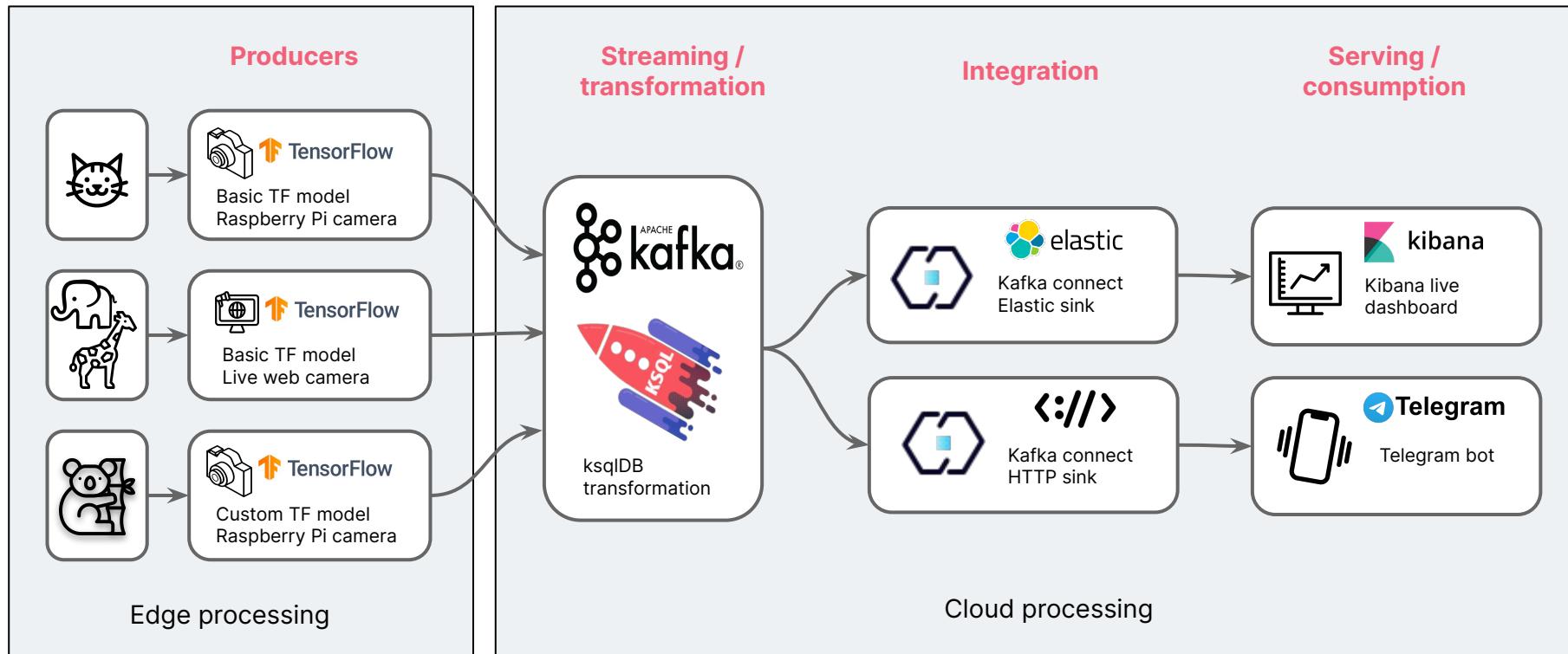
Telegram API

Chat ID

Regular expression

Architecture

Final version





5. How does this help me?



Events are everywhere



Photo by [Michał Parzuchowski](#) on [Unsplash](#)

Coffee queue **wait times**



Photo by [Donald Giannatti](#) on [Unsplash](#)

Shopping trolley **usage**



Photo by [Christian Wiediger](#) on [Unsplash](#)

Car park **occupancy**

Understand **streams that matter** - object detection plus stream processing



Photo by [Marcus Wallis](#) on [Unsplash](#)

That which is measured improves.

That which is measured and reported improves exponentially

Karl Pearson

Thanks / Any questions?

The screenshot shows a Confluence page with the following details:

- Header:** CONFLUENT, PRODUCTS, SOLUTIONS, LEARN, DEVELOPERS, GET STARTED FREE
- Title:** Real-Time Wildlife Monitoring with Apache Kafka
- Author:** SIMON AUBURY
- Date:** AUGUST 15, 2022
- Text:** Wildlife monitoring is critical for keeping track of population changes of vulnerable animals. As part of the Confluence Hackathon 2021 I was inspired to investigate if a streaming platform could help with tracking animal movement patterns. The challenge was to examine trends in identified species and demonstrate how animal movement patterns can be observed in the wild using Apache Kafka® and open source dashboards.
- Diagram:** A flow diagram illustrating the data pipeline. It shows an elephant and a giraffe icon connected to a camera icon, which is connected to a laptop icon labeled "kafka". The laptop icon is connected to a monitor icon and a smartphone icon.



Photo by [David Clode](#) on [Unsplash](#)

