SOBIGDATA.it
ITALIAN RESEARCH INFRASTRUCTURE

Finanziato dall'Unione europea NextGenerationEU | Ministero dell'Università e della Ricerca | Italiadomani | SOBIGDATA | Consiglio Nazionale delle Ricerche

Insights from Image Processing

Freeda's experience

FREEDA

Pisa, 27/03/2025

Why we are here

- We are NOT here to show you the latest advancements in Generative AI or to explain the theory behind cutting-edge technology… sorry
- We are here to show the harsh reality of adapting technology in an industry setting:
  - Trials and errors
  - The real data nightmares
  - Coming up with heuristics
  - Coping with the process

Outline

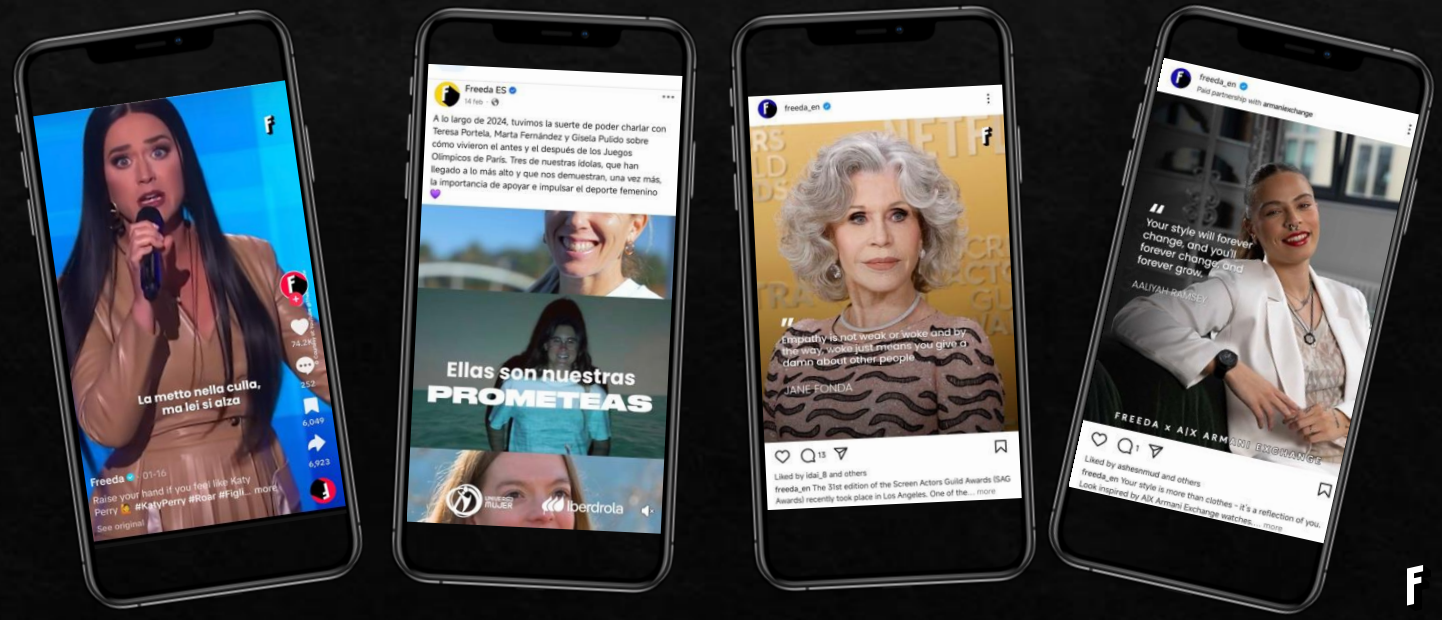
- Freeda
- Image & video classification
- Data challenges, heuristics, results from a real case study
- Engagement prediction
- Final remarks

Freeda

Who we are & what we do

Freeda Media & Freeda Platform

- Freeda starts out as a *digital media publisher* (Freeda Media)
  - We produce contents to be published on our own social media pages addressing themes relevant for Millenials and Gen Z, especially women and individuals feeling underrepresented
  - Alongside the editorial content, we curate the *branded content* built in collaboration with clients who want to reach our audience

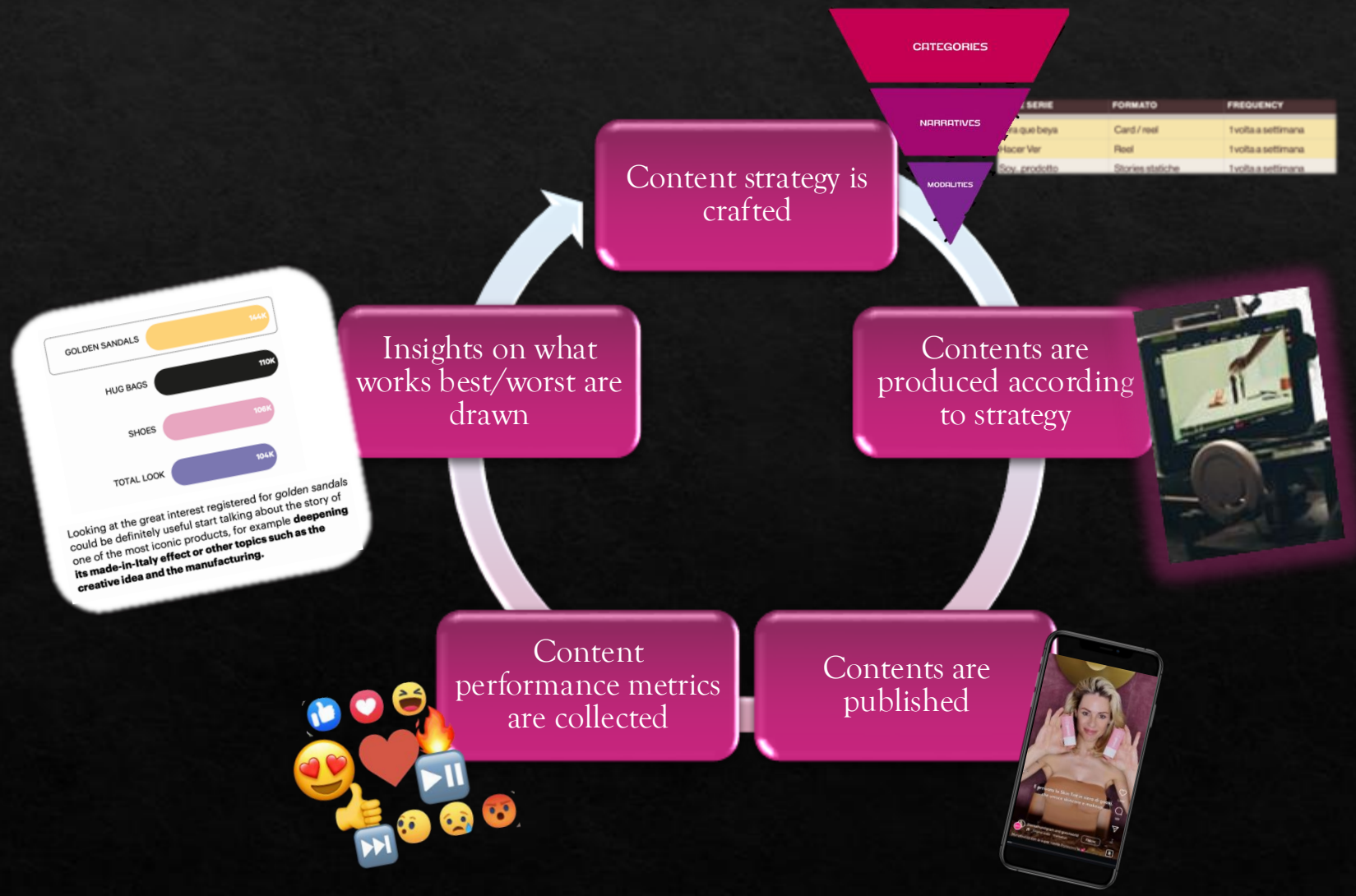


Freeda Media & Freeda Platform

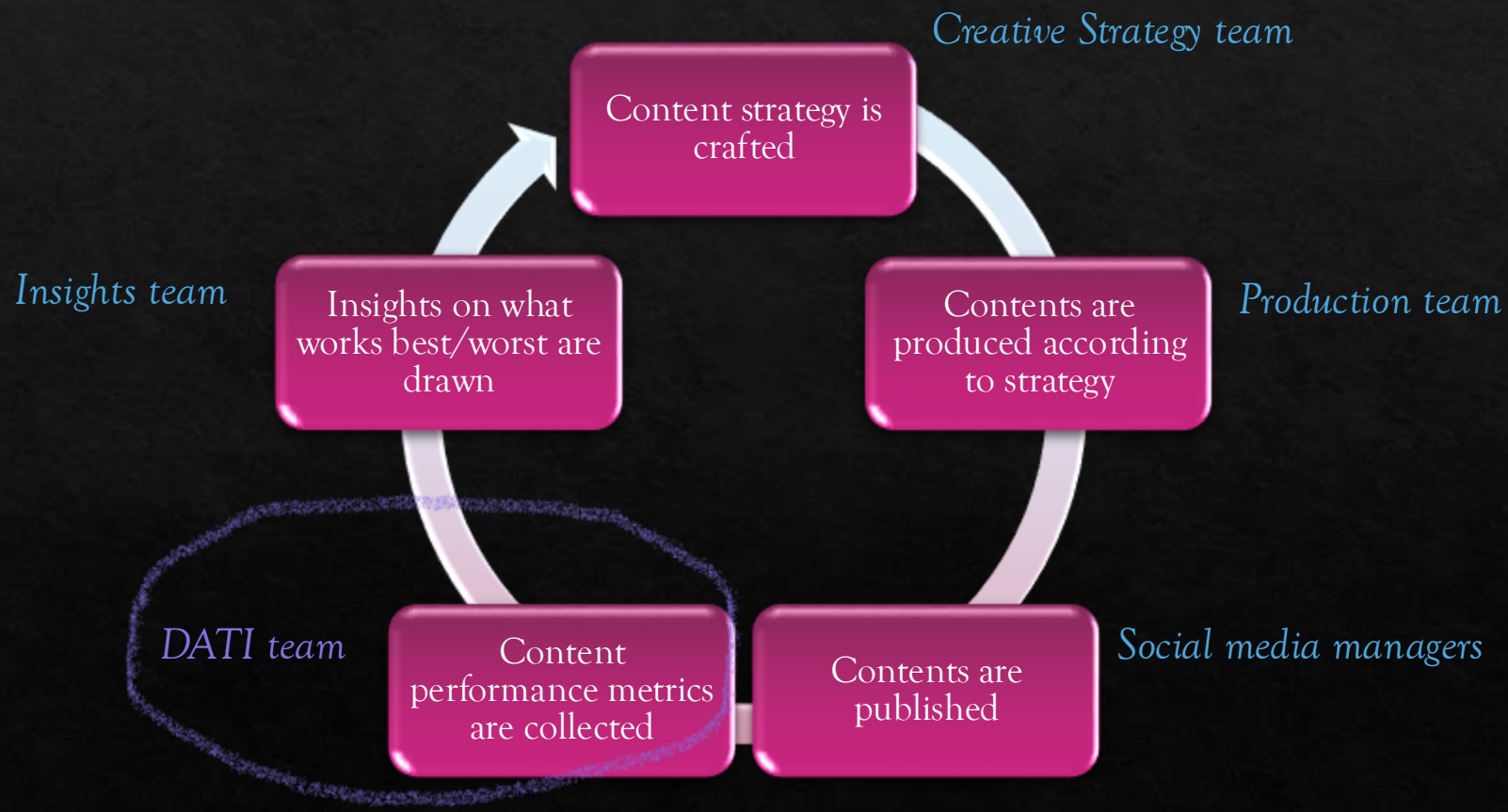
- More recently, a new division has been developed, Freeda Platform, acting as an *advertising agency* for brands
  - We support clients in evaluating and improving their communication (*listen services*), help them create engaging contents for their channels (*engage service*), and following the user-to-customer journey (*transact service*)

Freeda in numbers

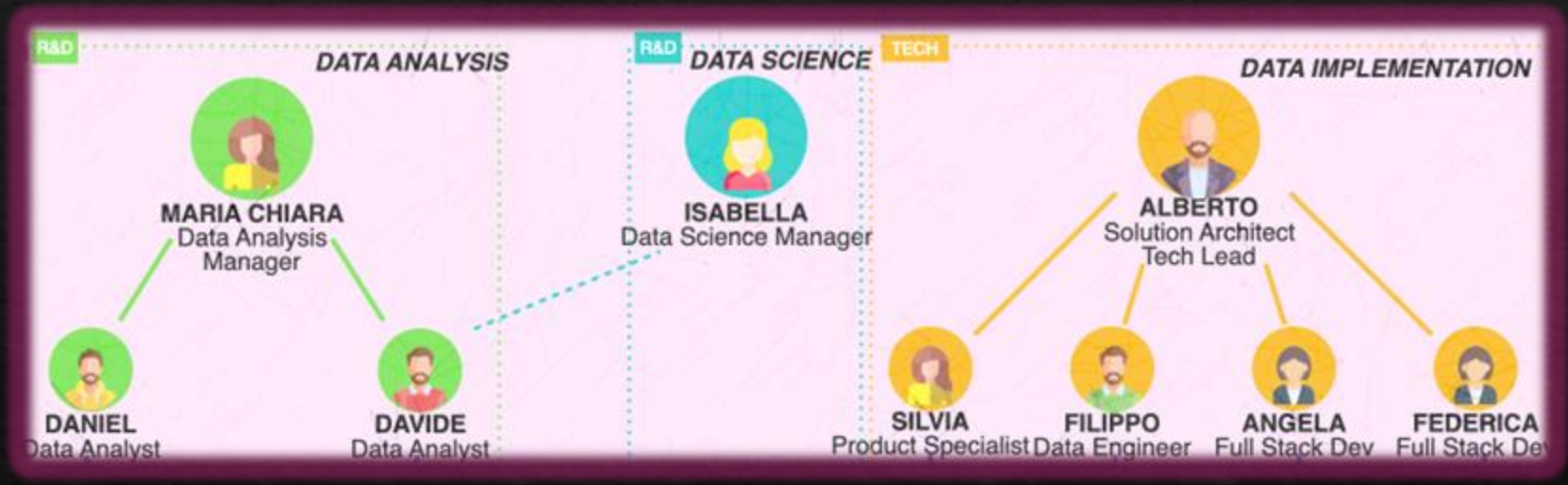
Data & Creativity



Data & Creativity



Data Analytics & Technology Innovation
DATI Team



- As a team, we take care of everything about data: from raw data collection to its storage, transformation & enrichment, from analysis to efficient presentation, e.g.:
  - ETL processes to get thousands of metrics data per day from social media platform APIs
  - Internal applications to store contents before publication, and explore their performance after
  - Customized dashboards & advanced analysis, also in support of the Insights team
  - Development & deployment of ML models to understand the relations between content features and performance

What does image processing have to do with this?

- The **core question** we wish to answer:

What makes a content successful on social media?

- **Idea**: decompose a content into a set of features and try to estimate the impact of each on observed performances

vs

?

- Features can be manually tagged or… automatically extracted! Here comes ML

Image & Video classification

Approach fundamentals

Image classification - basics

- (Image) classification is one of the fundamental ML tasks
  - Given a set of predefined classes (labels), the classifier must decide which class/es the input belongs to

 → *{'cat', 'dog'}?*

  - Supervised learning: the model is trained on pairs of <input, target> examples so that it can learn how to *generalize* their relation to unseen examples

 *, 'cat'>*  *, 'cat'>*  *, 'dog'>*

  - Many models/algorithms exist to tackle this task → for image classification, Convolutional Neural Networks (CNNs) are considered to be particularly efficient
    - Characterized by «convolutional layers» that filter images to progressively extract their features (e.g. edges)

Image classification – Model training

- Training a neural network means repeatedly presenting training examples to the model, observing the predicted ouput, and adapting the network parameters (its weights) so that on the next runs the prediction error (loss) gets lower
  - Large, labelled datasets are needed!
  - New models from the scientific community are customarily trained on the *ImageNet* dataset (1000 classes, 1.2M training examples in the most commonly used subset)

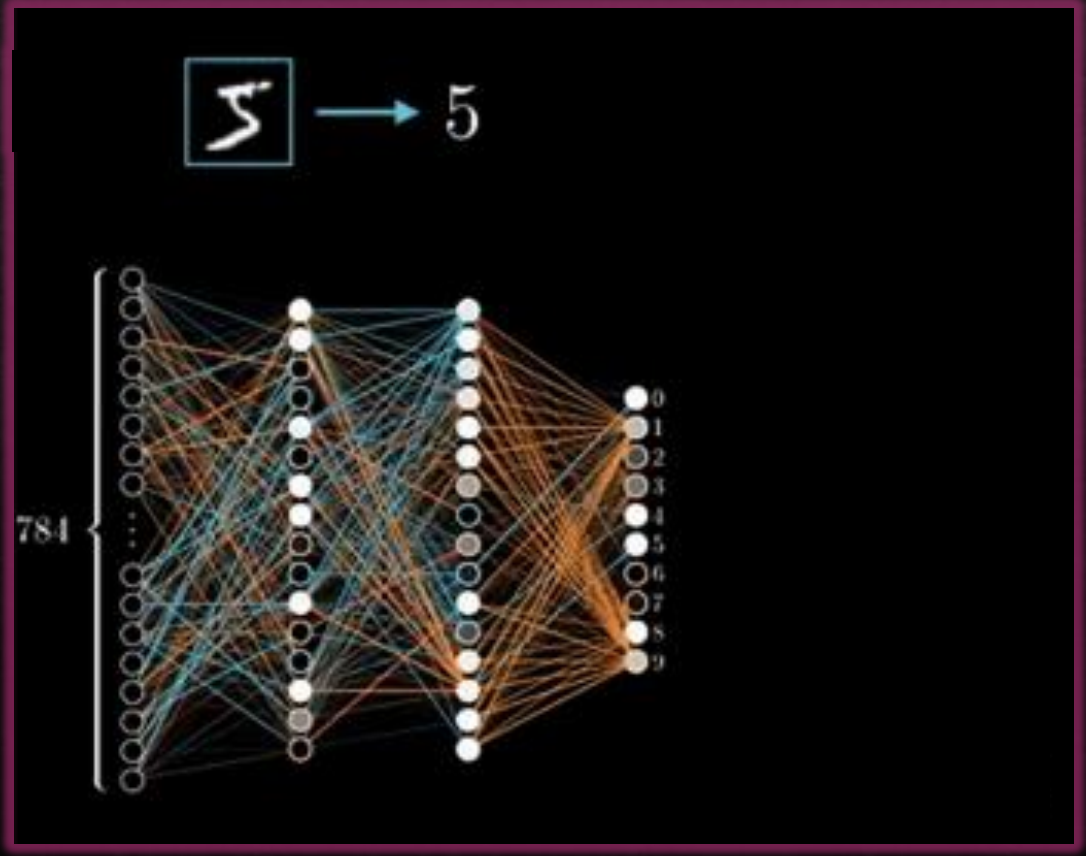


Image classification – Transfer Learning

- Training a deep neural network from scratch can be very expensive/impractical
  - In terms of dataset preparation and computational resources
- Enter Transfer Learning!
  - Idea: use a pre-trained model as a starting point and continue training on a «small», tailored dataset
  - This fine-tuning process can both take advantage of the general knowledge on visual features acquired by the model during pre-training and specialize it to recognize the classes we are interested in
  - E.g. *PyTorch (torchvision)* & *Huggingface* make several pre-trained models available that can be used for transfer learning

Transfer Learning in practice: decisions to be made

1. Which classes do I want to recognize?
2. What kind of images do I have/do I want to use for training?
3. Which image transformations make sense for my task?
4. Which pre-trained model(s) do I want to fine-tune? With which hyperparameters?
5. Which accuracy level do I consider «good enough» to move to production?

Transfer Learning in practice: video classification

- We can adapt pre-trained/fine-tuned image classifiers to perform video classification as well
  - We can sample a number of frames from the video, and classify each frame indipendently
  - … and then aggregate the frame labels into one label at the video level

- … Which adds a new set of decisions to be made:
  1. How many frames should I sample? And how should I sample them?
  2. How do I aggregate frame-level labels into a video-level one?

Spoiler: there is no predefined answer.

You need to find them by getting your hands dirty with actual data!

Content classification: case study

Data Challenges, Heuristics, and Results

Classifying people presence in images/video

- *Motivating observation:* contents showing people usually perform better than contents with no people
  - … but does the actual number matter, too?

1. Which classes do I want to recognize?

- To avoid having too many, fine-grained labels, we decided to classify images into 4 classes:
  - *[no person, one person, two persons, three or more persons]*
- Rationale:
  - Too many labels may require a larger labelled dataset (so that all classes are «equally» represented)
  - At inference time, we might expect to have underrepresented classes → sparse data make it hard to draw robust conclusions
  - Considerations stemming from business knowledge & experience with the data domain

Dataset preparation

- Data retrieval: how many data points do we need?
  - No obvious answer: our heuristic is around 2K
- Data labelling: is it *that* straightforward?

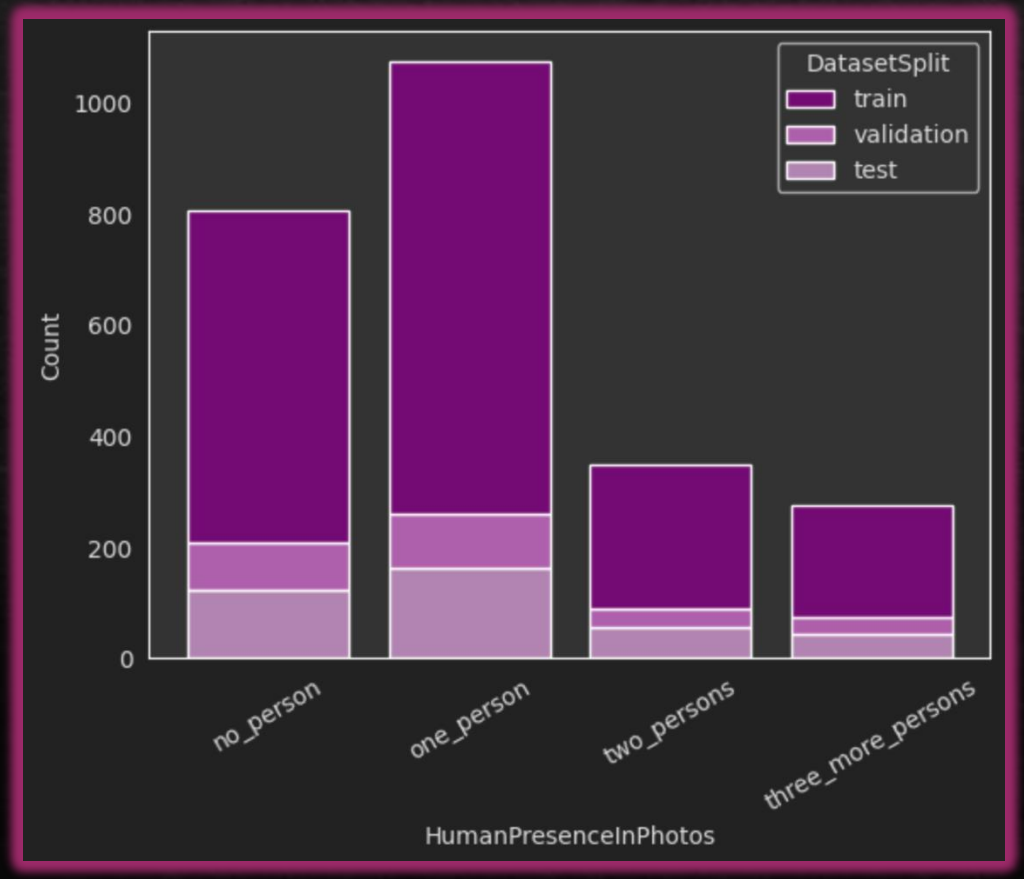
Dataset preparation

2. What kind of images do I have/do I want to use for training?

- Caveat: look at your data like you were an image classifier!
  - Visual features differ a lot between photos and illustrations
  - Training a classifier on such different data distribution may harm its overall accuracy
  - It makes more sense to train separate classifiers to optimize their performance on specific subsets of images

Final dataset

- 2502 images:
  - Records in training set = 1873 (70%)
  - Records in validation set = 250 (10%)
  - Records in test set = 379 (20%)
- Slightly imbalanced but still acceptable



Data augmentation

- Data augmentation is a standard practice to increase the overall number of training examples fed to a model, while effectively keeping the same dataset
  - Perturbed versions of the original data are obtained through *transformations*
  - This improves model accuracy and generalization
    - It's like looking at the same thing from different angles/with different eyeglasses

Rotation

Translation

Solarization

Data augmentation

3. Which image transformations make sense for my task?

- No obvious answer! Experimentation is needed
  - Some transformations are always needed:
    - normalization (usually using the mean and std computed on the original ImageNet dataset)
    - resizing (to a min/max image size required by the chosen model)
  - Some transformations can be detrimental for your task/dataset:
    - If you want to detect text presence in your images, and text often occurs near the borders of the image, using a Crop transform may be risky!
  - Data augmentation is usually performed on the training set only
    - as we want to evaluate the model on the original versions of our validation/test images

```
img_transforms = {
'train': transforms.Compose([
transforms.Resize(input_size),
transforms.TrivialAugmentWide(),
transforms.CenterCrop(size=input_size),
transforms.ToTensor(),
transforms.Normalize(IMAGENET_MEAN,IMAGENET_STD)
]),
'validation':transforms.Compose([
transforms.Resize(input_size),
transforms.CenterCrop(size=input_size),
transforms.ToTensor(),
transforms.Normalize(IMAGENET_MEAN,IMAGENET_STD)
]),
'test':transforms.Compose([
transforms.Resize(input_size),
transforms.CenterCrop(size=input_size),
transforms.ToTensor(),
transforms.Normalize(IMAGENET_MEAN,IMAGENET_STD)
])
}
```

Model training

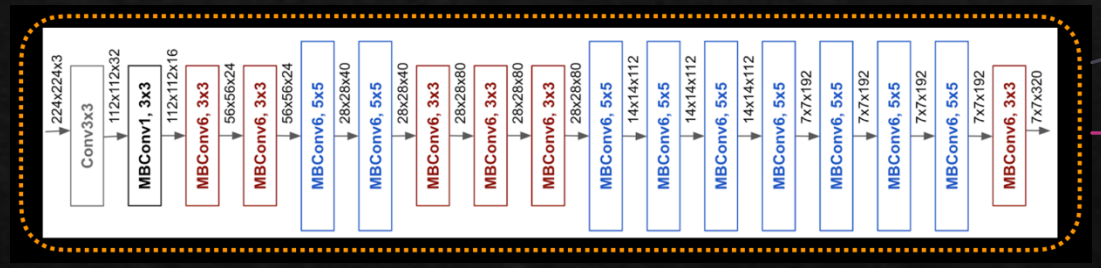
4. Which pre-trained model(s) do I want to fine-tune? With which hyperparameters?

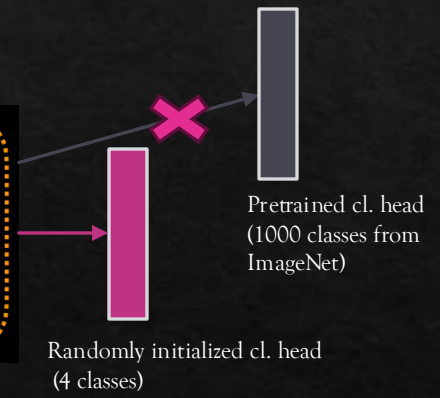
- Guess what? No obvious answer! Experimentation is needed
  - Usually, resnet101 and efficientnet_v2_m perform well
  - TorchVision also offers alternative sets of pre-trained weights for some architectures

```
training_parameters =
{'CLASS_TASK': 'HumanPresenceInPhotos',
'FEATURE_EXTRACT': False,
'WEIGHTS': 'DEFAULT',
'MODEL_TYPE': 'efficientnet_v2_m',
'BATCH_SIZE': 16,
'USE_CUDA': True,
'SEED': 10,
'learning_rate': 0.002,
'momentum': 0.9,
'step_size': 7,
'gamma': 0.1,
'epochs': 20 }
```

Model training

4. Which pre-trained model(s) do I want to fine-tune? With which hyperparameters?






```
training_parameters =
{'CLASS_TASK': 'HumanPresenceInPhotos',
'FEATURE_EXTRACT': False,
'WEIGHTS': 'DEFAULT',
'MODEL_TYPE': 'efficientnet_v2_m',
'BATCH_SIZE': 16,
'USE_CUDA': True,
'SEED': 10,
'learning_rate': 0.002,
'momentum': 0.9,
'step_size': 7,
'gamma': 0.1,
'epochs': 20 }
```

- You can choose to train only the classification head (a.k.a. using the pretrained model as a feature extractor) or to fine-tune the whole model
  - We got better results with the latter approach

Model training

4. Which pre-trained model(s) do I want to fine-tune? With which hyperparameters?

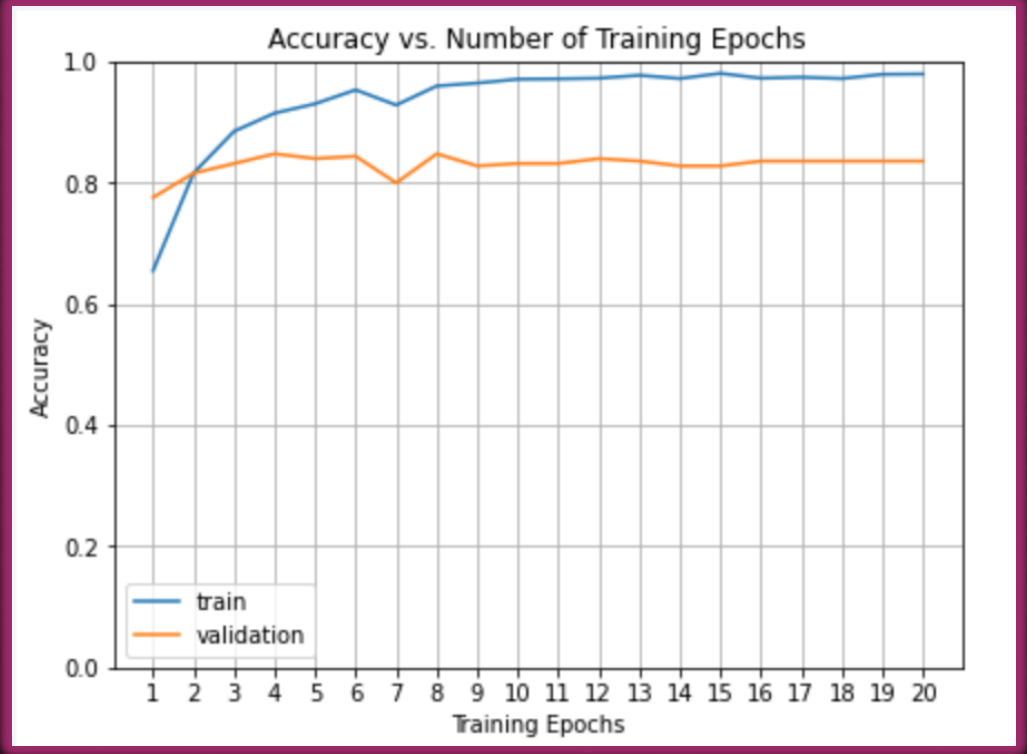
- No fixed recipe for training regimen either
  - Here we got the best results with an SGD optimizer and StepLR scheduler
  - Learning rate and momentum are probably the hyperparameters that affect model performance the most
  - 20 epochs (with early stopping) are usually enough to get good results on similar tasks/dataset sizes
  - A larger batch size speeds up training but make sure it fits into available memory!
  - GPUs definitely help
  - Setting the random seed allows for reproducibility (logging parameters for each run is wise, too)

```
training_parameters =
{'CLASS_TASK': 'HumanPresenceInPhotos',
'FEATURE_EXTRACT': False,
'WEIGHTS': 'DEFAULT',
'MODEL_TYPE': 'efficientnet_v2_m',
'BATCH_SIZE': 16,
'USE_CUDA': True,
'SEED': 10,
'learning_rate': 0.002,
'momentum': 0.9,
'step_size': 7,
'gamma': 0.1,
'epochs': 20 }
```

Model evaluation

5. Which accuracy level do I consider «good enough» to move to production?

- In real life, if you get 99% accuracy you must get *very* suspicious
  - > 80% (on validation set) is already «good enough»
  - Error analysis is crucial to highlight potential model biases

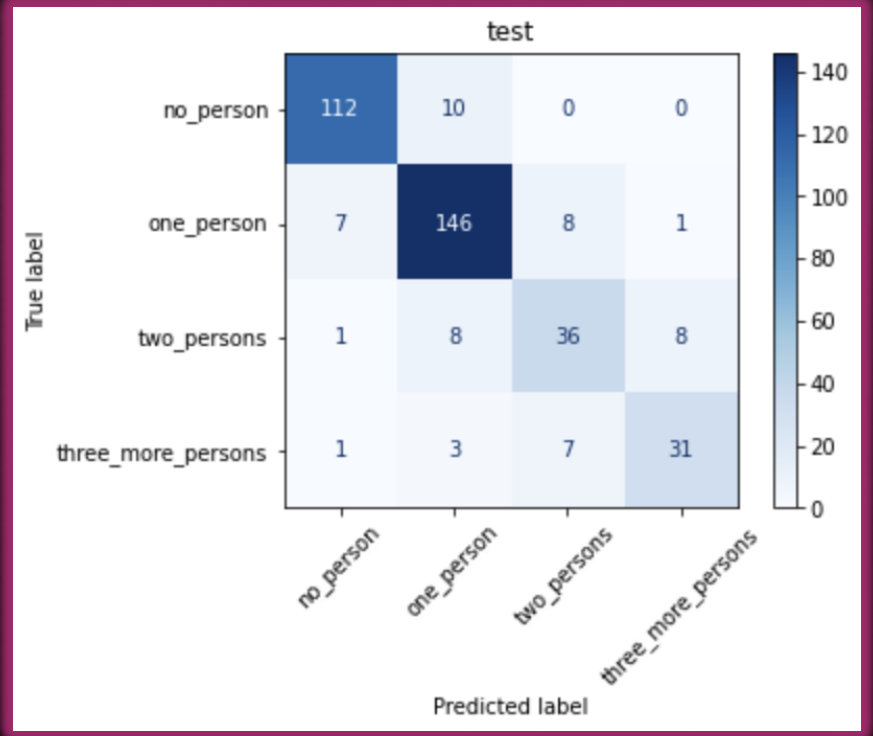


{"accuracy_split": {"train": 0.974, "validation": 0.848, "test": 0.858}

Model evaluation

5. Which accuracy level do I consider «good enough» to move to production?

- Does your model make «reasonable mistakes»?



Label: no_person
Prediction: one_person
Probability: 0.948

Label: two_persons
Prediction: three_more_persons
Probability: 0.802

Label: three_more_persons
Prediction: two_persons
Probability: 0.836

Label: one_person
Prediction: two_persons
Probability: 0.990

Moving from images to videos

- The good news is that you can re-use a model trained for images to also classify videos…
- The bad news is that you will need to handle videos (hello *ffmpeg!*)

Moving from images to videos

6. How many frames should I sample? And how should I sample them?

- You need to find a balance between video coverage and computational load
- We developed a heuristic sampling strategy
  - To reduce memory usage, we first scale down frame size
  - We consider only the first 5 minutes of a video (most users will not even watch that far into the content)
  - We sample a maximum of 12 frames per video, choosing one representative frame (no more than) every 2 seconds

```
FRAME_EXTRACTION_PARAMS:
    MAX_SIZE: 1024
    FILTER: 'THUMBNAIL_EVERY_N'
    SAMPLING_MODE: 'BOUNDED_PERIODIC_SAMPLING'
    MAX_VIDEO_LENGTH: 300
    MAX_N_FRAMES: 12
    EVERY_N_SECONDS: 2
```

Moving from images to videos

7. How do I aggregate frame-level labels into a video-label one?

```
AGGREGATION_PARAMS:
    AGGREGATION_MODE: 'ORDINAL'
    MIN_PERC_FRAMES: 0.25
```

- Which is the best aggregation strategy? No single answer!
  - Most common label? Reasonable, but not necessarily correct (think binary classifiers)
  - For the selected task, we choose to adopt the frame-level label corresponding to the highest number of people
    - Provided it is assigned to at least 25% of the frames (to filter out noise/errors)

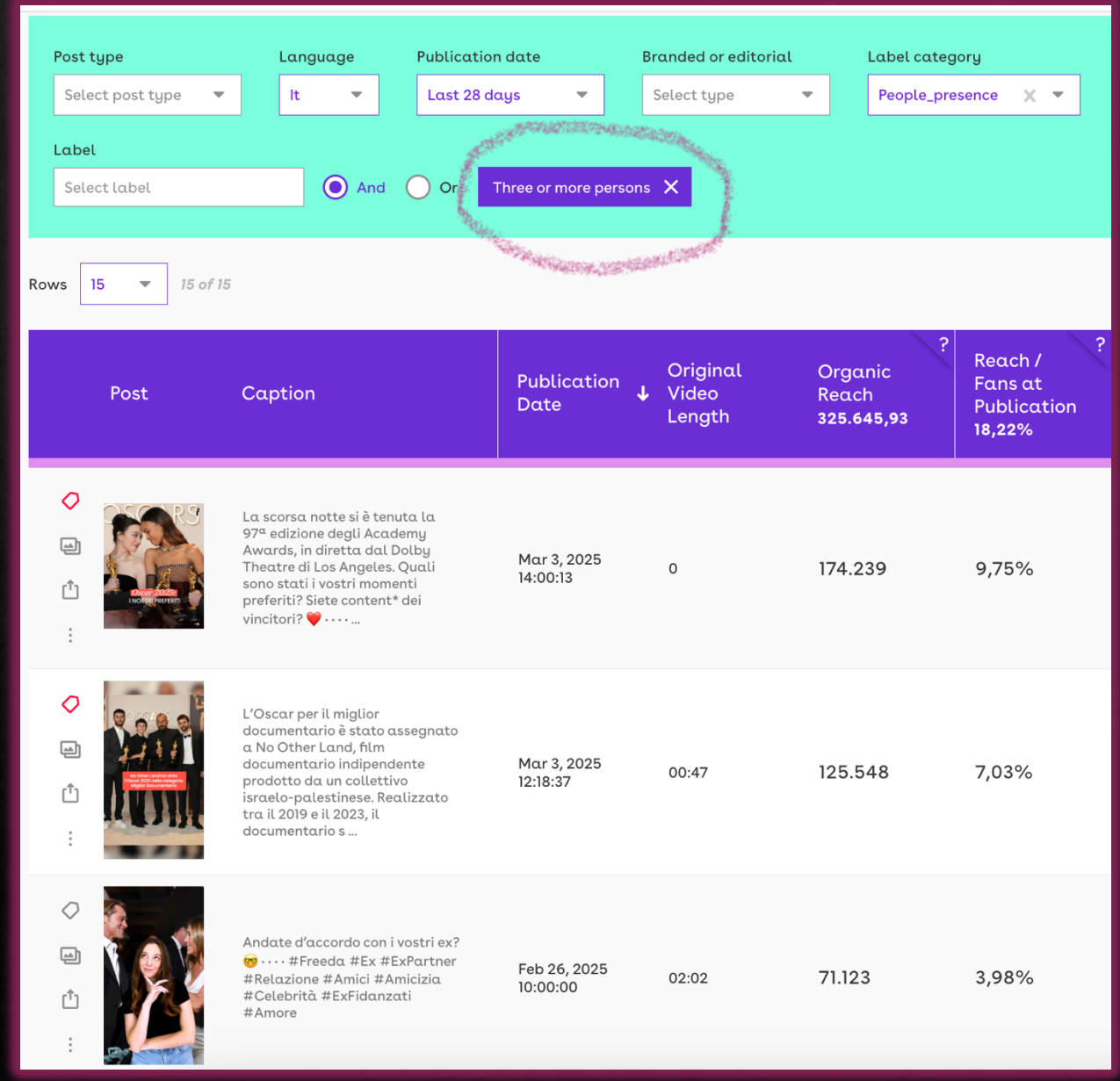
Final video label: «Two persons»

One last, easy question…

- How do I deploy my models?
  - *Jupyter notebooks is where ML models go to die*
  - Where & how will I get data from? Where & how will I write data to? When & where will my model run?
- Lots of MLOps tools/options out there to choose from, based on your needs
  - We chose Valohai for experiment tracking, model serving
  - We set up 23 pipelines made of a data ingestion step, a computation step (model inference), and a result persistence step
  - Scheduled jobs run every night to process new files in our content store
  - Ideally, a user feedback-retraining loop should also be implemented to keep models accurate

Content classification: the end of the journey

- Now, our automatic labels can be served in our internal applications and retrieved from our data lake!
- Next step: use labels to study how content performance varies across groups of contents sharing the same feature(s)

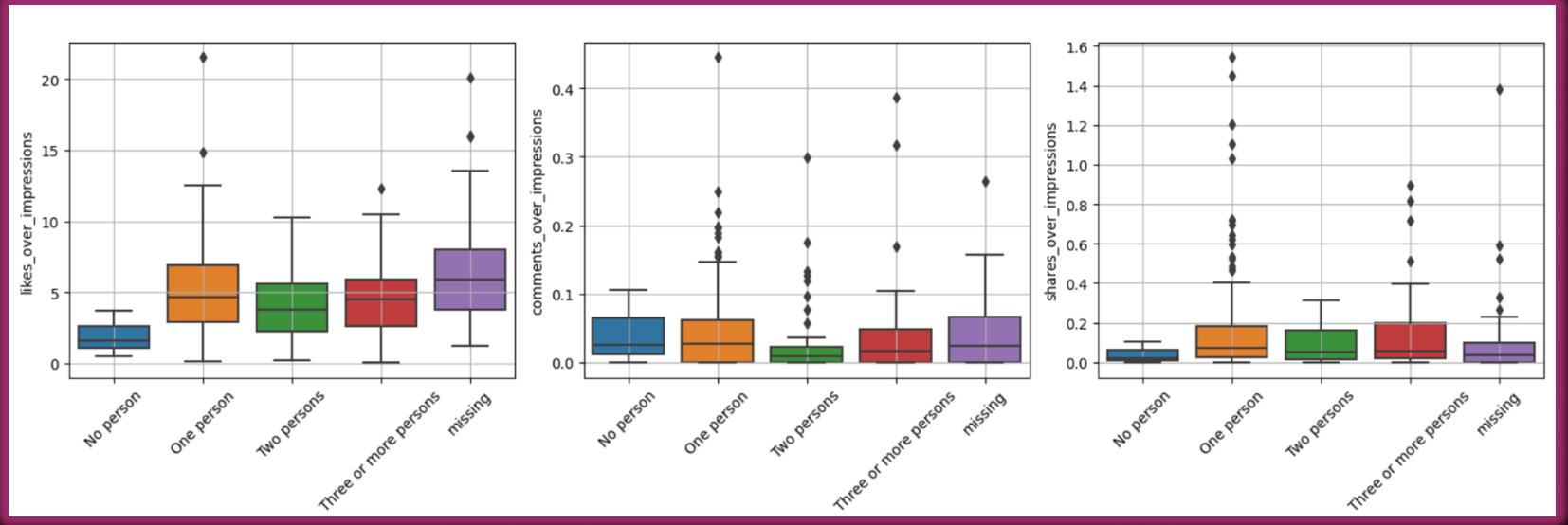


Engagement prediction

Using labels to study content quality/performance

From Features to Engagement

- Basic assumption: a well-constructed content will receive more likes/comments/shares (*engagement* ↑)
  - and in turns it will be shown to more people (*reach* ↑)
- So, which features make up good content?



TikTok contents published by 8 publishers during a 14-months timeframe (n = 1204)

From Features to Engagement

- … but a content is made of a **combination** of features!
  - Feature importance: which features are really responsible for the success of that one content?
    - Do contents with people work better per se, or do they maybe incidentally always include a cat, too?
  - Interactions: which combinations work well?
    - What if contents with people work well only when the post has an ispiring tone of voice, and contents with no people only when the post has an informative tone of voice?
- We *just* have one problem: data sparsity

Label data – behind the scenes

- Contents are both labelled by our automatic classifiers and by manual taggers from other teams
  - Manual labels cover higher-level concepts, such as «tone of voice/objective», «temperature», or «genre»
- Manual labels suffer from:
  - Subjectivity issues (taggers may not agree on how to label a content)
  - Missing data (taggers can forget to label contents)
  - Proliferation (taggers can use «free keywords» that rapidly get out of control)
- Still, they can convey important content characterization

The curse of dimensionality

- Large number of label categories + high cardinality of labels within categories (proliferation) → explosion of the feature space
  - We have around 30 label categories (automatic + manual) and some of them include thousands of labels!
- It gets basically impossible to collect enough data to adequately cover such a space → data become sparse
  - Overfitting
  - Low reliability of prediction
- Missing data exacerbates the problem
  - You may be forced to discard incomplete records

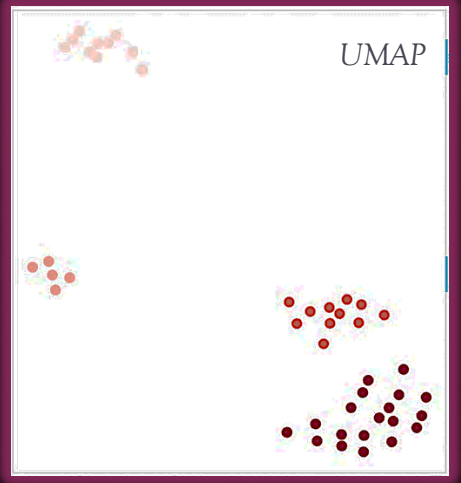
The curse of dimensionality & dataset sparsity

	FORMAT_CATEGORY_CAROUSEL	FORMAT_CATEGORY_VIDEO	FORMAT_LABEL_Carousel Photos	FORMAT_LABEL_Video Original	TOPIC_Beauty	TOPIC_Body
0	0	1	0	1	0	0
1	0	1	0	1	0	0
2	0	1	0	1	0	0
3	0	1	0	1	0	0
4	0	1	0	1	0	0
5	0	1	0	1	0	0
6	0	1	0	1	0	0
7	0	1	0	1	0	0
8	0	1	0	1	0	0
9	0	1	0	1	0	0
10	0	1	0	1	0	0
11	0	1	0	1	0	0
12	0	1	0	1	0	0
13	0	1	0	1	0	0
14	0	1	0	1	0	0
15	0	1	0	1	0	1
16	0	1	0	1	0	1
17	0	1	0	1	1	0
18	0	1	0	1	1	0
19	0	1	0	1	1	0
20	0	1	0	1	1	0
21	0	1	0	1	1	0
22	0	1	0	1	1	0
23	1	0	1	0	0	0

...

OBJECTIVE_Inspiring	OBJECTIVE_LOL	TEMPERATURE_Cold	size
1	0	1	1
1	0	1	1
0	0	1	1
1	0	1	2
0	0	1	1
0	0	1	1
0	0	1	5
0	1	1	1
0	1	1	2
0	1	1	1
0	0	1	1
1	0	1	1
0	0	1	2
0	0	1	6
0	0	1	2
1	0	1	1
0	0	1	1
0	1	1	1
1	0	1	3
0	0	1	1
0	0	1	2
0	0	1	6
0	0	1	1
1	0	1	1

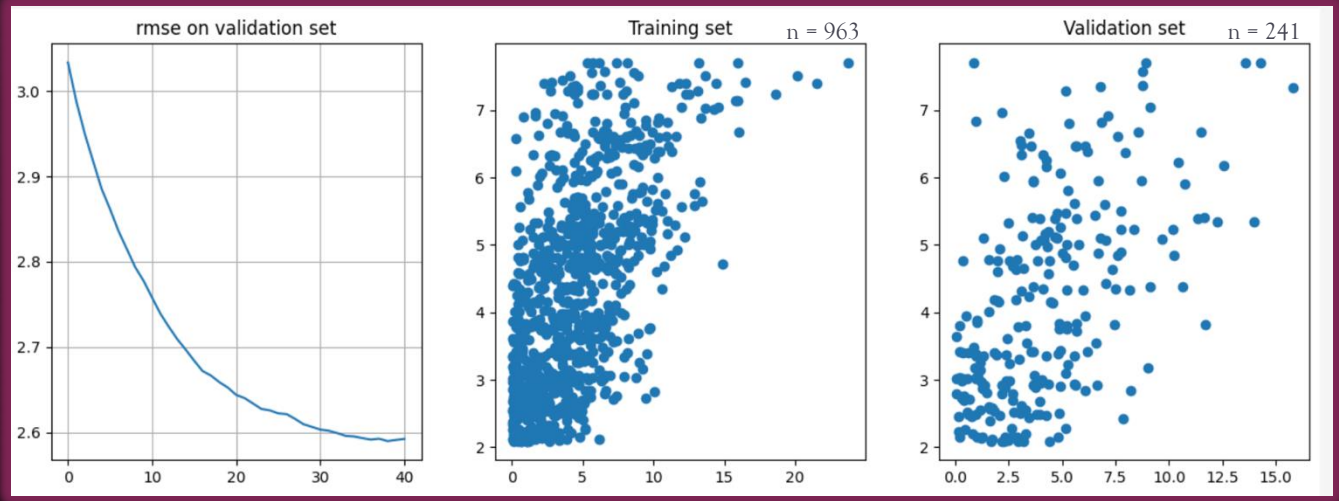
24 unique combinations over 45 records



- Feature selection (both based on data statistics and domain knowledge) may help to some extent
- Dimensionality reduction may help too, but will hinder interpretation

First attempts at a model for engagement

- We trained an XGBoost regression model on a subset of labels and other info (#features = 13) to predict *likes_over_impressions*
  - XGBoost (and similar) is still the go-to choice when working with numerical & categorical features
  - For the reasons above, and more (incomplete data representation), we cannot really expect good model performance
    - But it looks like there is some signal in the data, at least

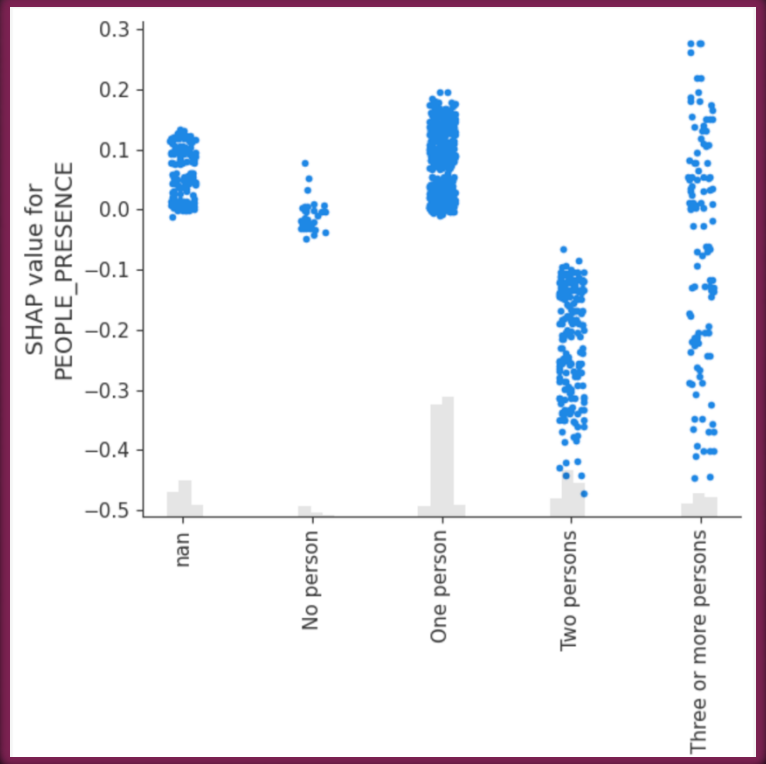


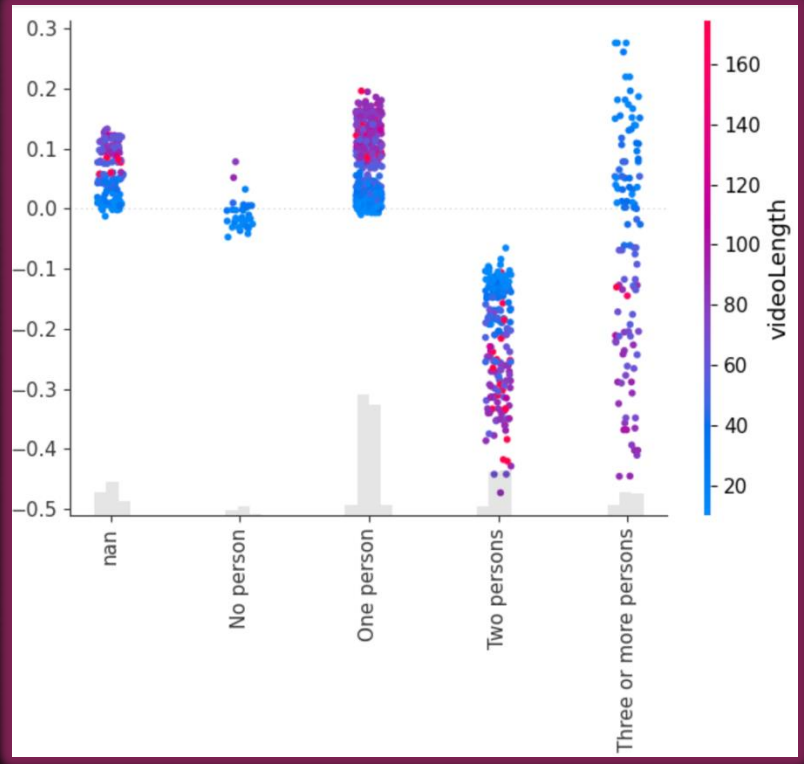
R2 - Training set: 0.340, Validation set: 0.296

First attempts at a model for engagement

- We can at least try to use the model to investigate which features impact (and how) on engagement

Feature	XGBoost importance score
FORMAT_LABEL	0.534
BRIGHTNESS	0.104
CONTRAST	0.063
videoLength	0.063
PEOPLE_PRESENCE	0.049
FORMAT_CATEGORY	0.038
BODY_FOCUS	0.031
OBJECTIVE	0.028
TATTOO_PRESENCE	0.025
TEMPERATURE	0.024
ANIMAL_PRESENCE	0.024
KISS_PRESENCE	0.016
frameCount	0.000





Final remarks

Coping with the process

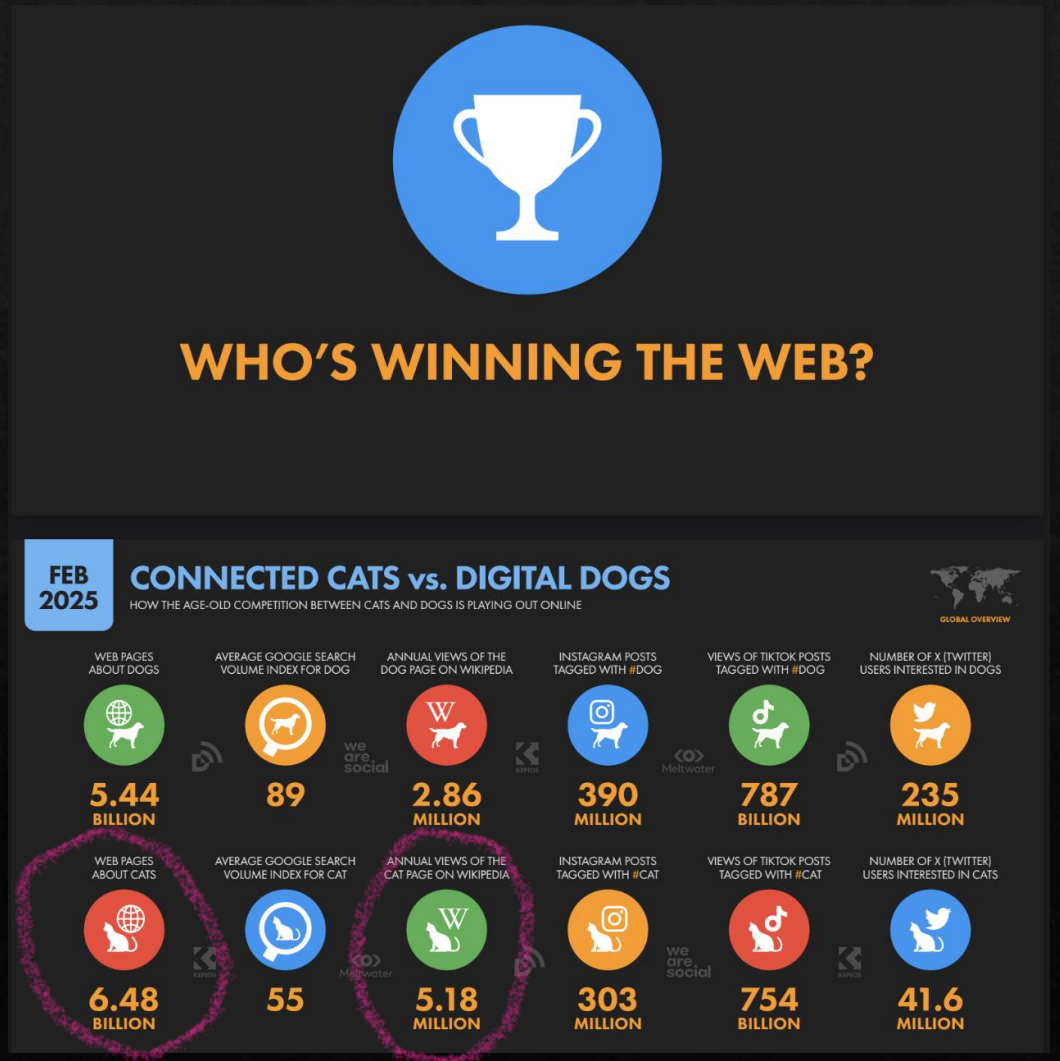
Image Processing is just one piece of the puzzle

- Everything starts from a single question: «what do we want to *measure* in our contents?»
  - The answer must be shared by all teams involved!
  - The answer may (and will) change in time: be ready to replace your classification models!
    - And if the label system changes (and it does) old data can become unusable for new engagement prediction models
  - Not everything will be (reliably) labelled by models (e.g. mood)
    - Manual labelling still needed → need for inter-team collaboration to minimize missing and incoherent labelling

A never ending endeavor

- Ideally, the engagement prediction model would be multimodal:
  - Running not only on categorical labels, but also on (embeddings of) sound, image/video, text → not a trivial model architecture
  - To fight the curse of dimensionality, we will need even more data!
  - We also experimented with different ideas (e.g. content similarity) that did not make it into production
- The way contents are produced and the way people respond to them on social media are continuously evolving
  - The data we are trying to model are constantly shifting, making it hard to collect enough coherent data points to train a model
- But are we going to stop trying? Probably not.

And in case you were wondering...



Thank you!

FREEDA

SOBIGDATA.it
ITALIAN RESEARCH INFRASTRUCTURE

Finanziato dall'Unione europea NextGenerationEU | Ministero dell'Università e della Ricerca | Italiadomani | SOBIGDATA | Consiglio Nazionale delle Ricerche