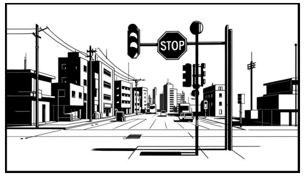
GEO-LOCATOR



→ ROME

THE ERDŐS INSTITUTE
Helping PhDs get jobs they love.
Helping you hire the PhDs you need.

FRANCESCA BALESTRIERI
ZACK BEZEMEK
DANTE BONOLIS
LEONHARD HOCHFILZER
AASHRAYA JHA

MOTIVATING PROBLEM

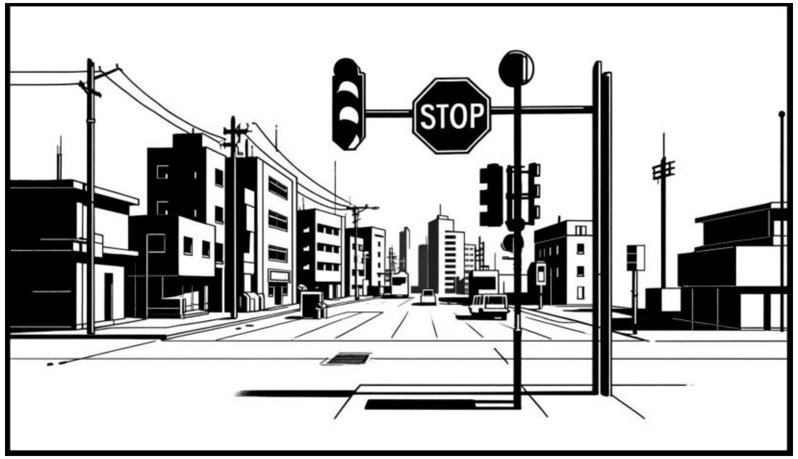
ROME OR MADRID?

MADRID OR ROME?

MADRID OR ROME?

ROME OR MADRID?

OUR APPROACH



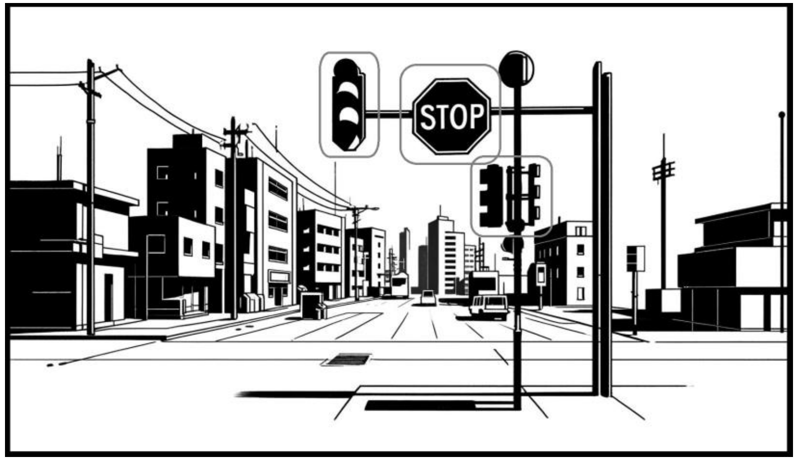
AN IMAGE IS QUITE
UNSTRUCTURED

ROME ???

EXPERT DOMAIN KNOWLEDGE

INSPIRED BY GEOGUESSR PROS:
they use **man-made** homogeneous/persistent **features highly specific to each country** as discriminants

BY ALSO CONSIDERING SPECIFIC FEATURES (IF AVAILABLE), WE IMPOSE **EXTRA STRUCTURE** ON THE IMAGE



THE DATASET

GSV-CITIES arxiv:2210.10239 / Neurocomputing 2022

- It contains ~530k images, across 23 different cities
- There are more than 62k different places, spread across multiple cities
- Each place is depicted by at least 4 images (up to 20 images)
- All places are physically distant (at least 100 meters between any pair of places)

EXAMPLE OF IMAGE METADATA:

	place_id	year	month	northdeg	city_id	lat	lon	panoid
0	1678	2014	10	370	Barcelona	41.402066	2.198988	DB4DzlzCRq4lyE9FMx_9Ow

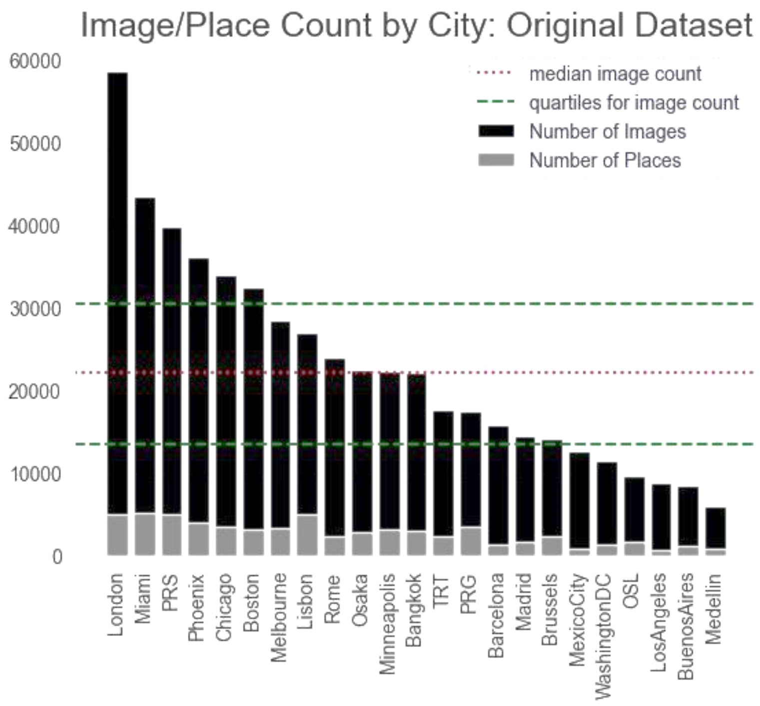
EXAMPLE OF A PLACE (BARCELONA, PLACE ID 17801):

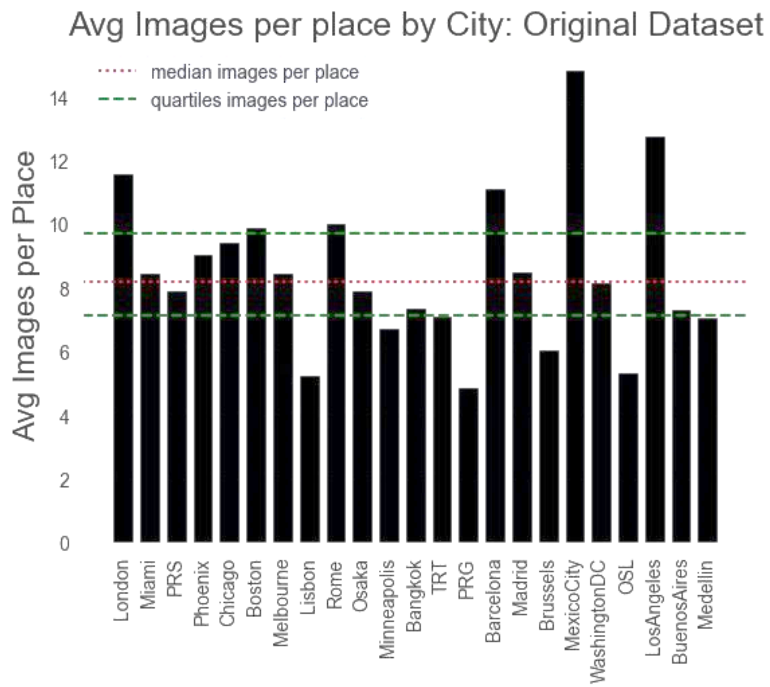
ORIGINAL DATASET

23 cities

Total number of images = **529506**

Total number of places = **64394**





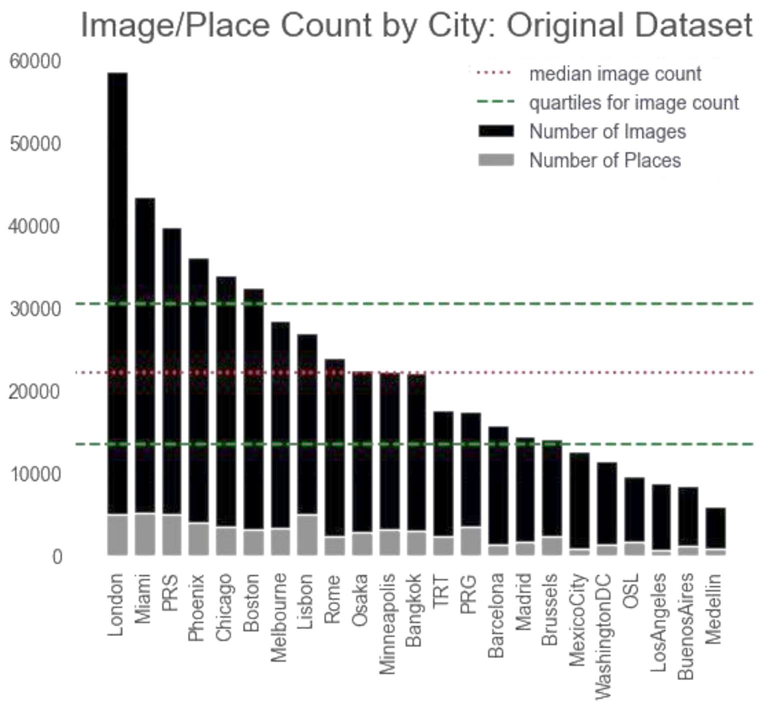
ORIGINAL DATASET

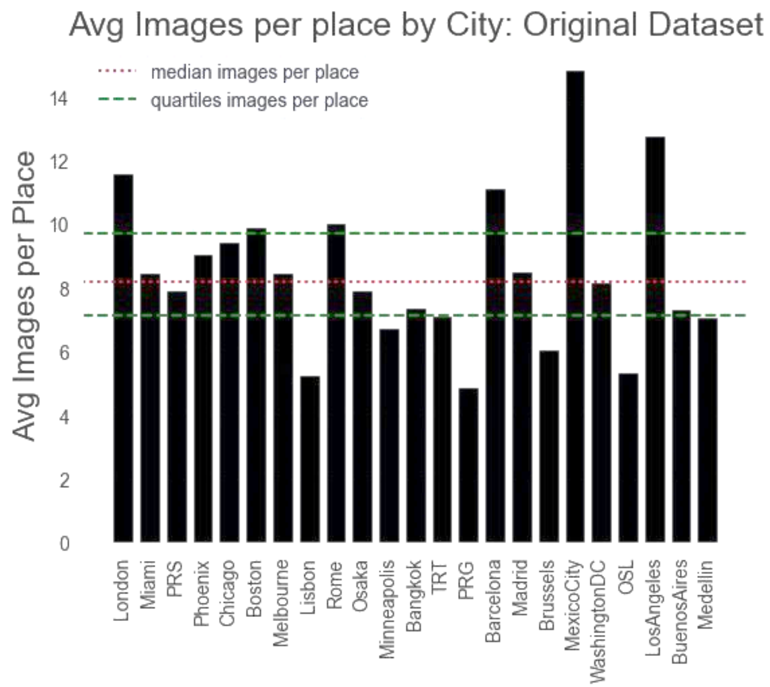
23 cities

Total number of images = **529506**

Total number of places = **64394**

CLEANING AND NORMALIZATION



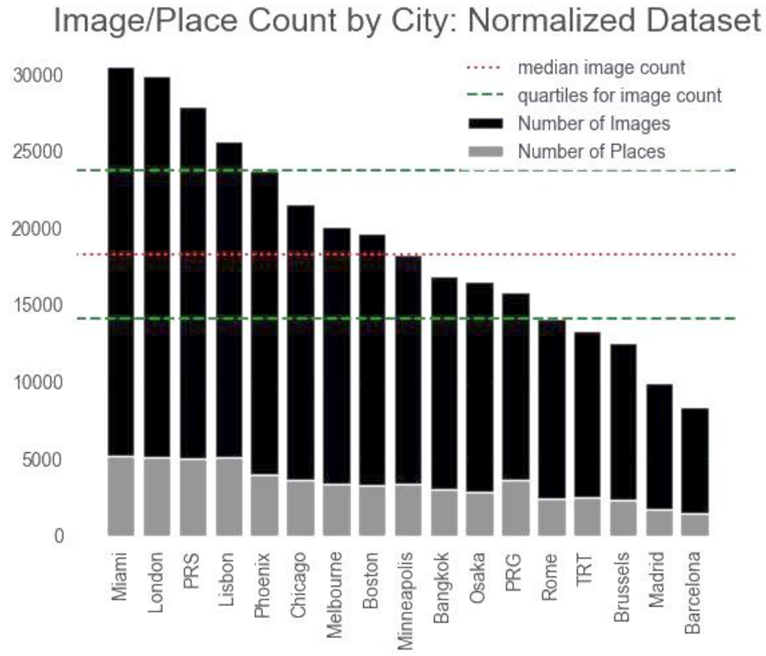


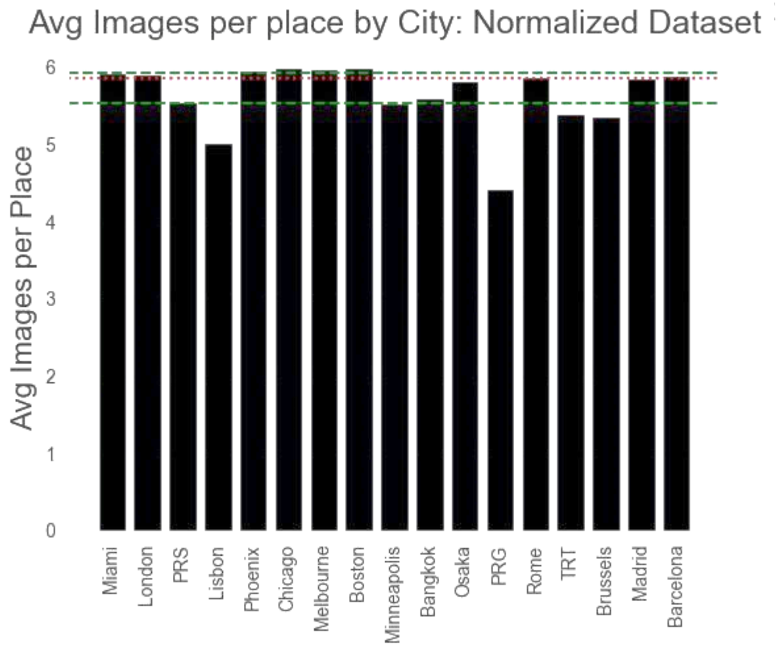
BALANCED DATASET

17 cities

Total number of images = **324697**

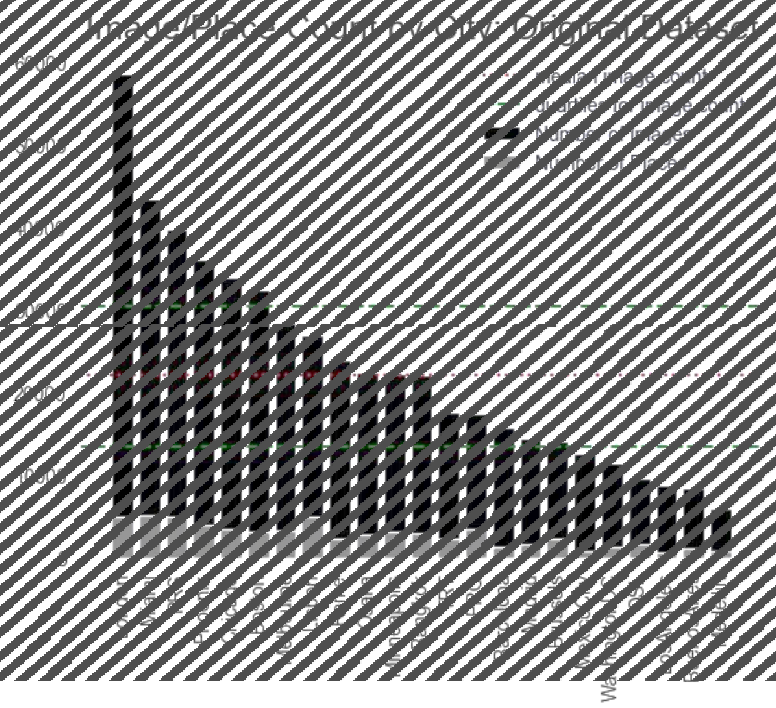
Total number of places = **57618**

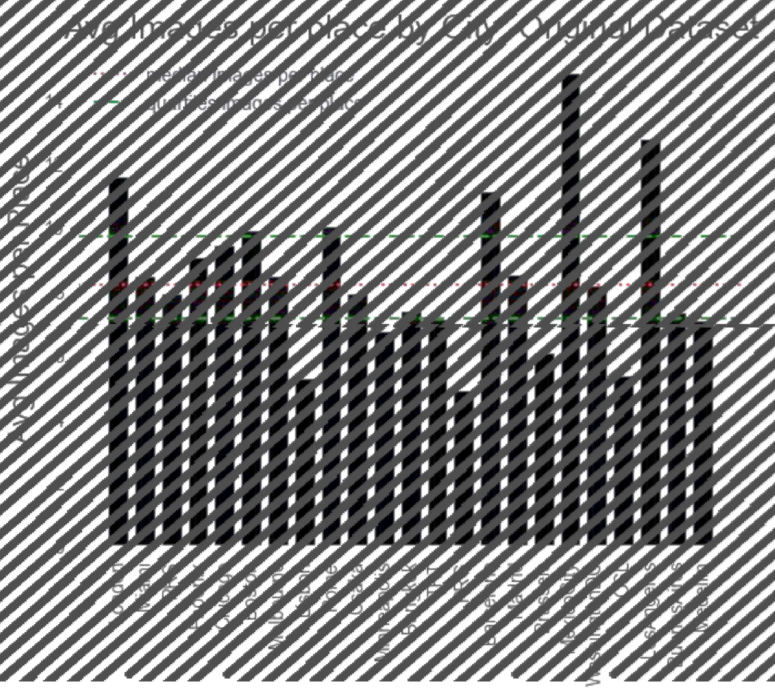




ORIGINAL DATASET

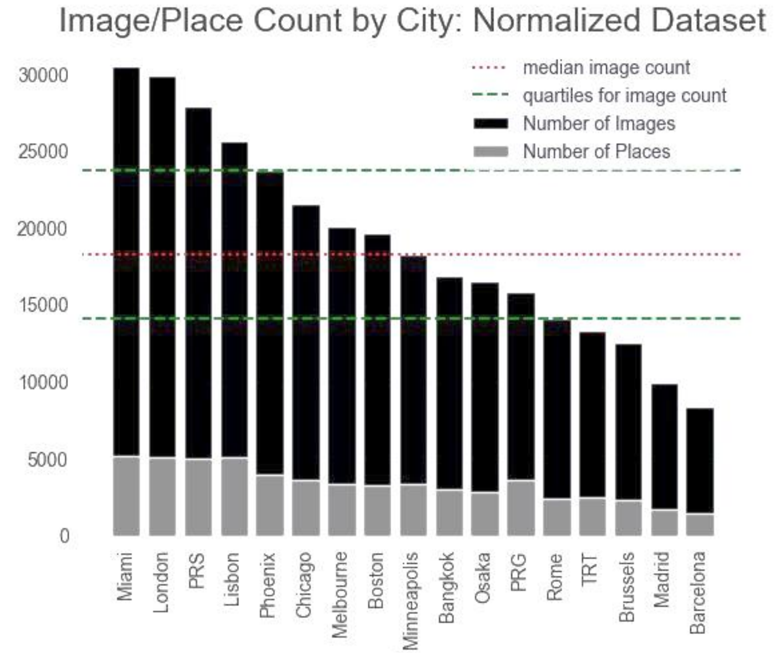
23 cities
Total number of images = **529506**
Total number of places = **84394**

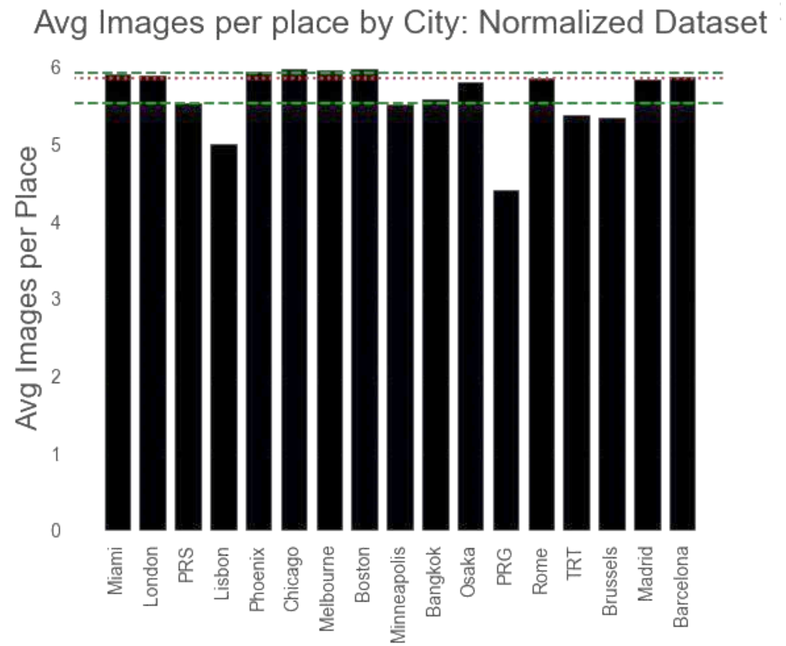




BALANCED DATASET

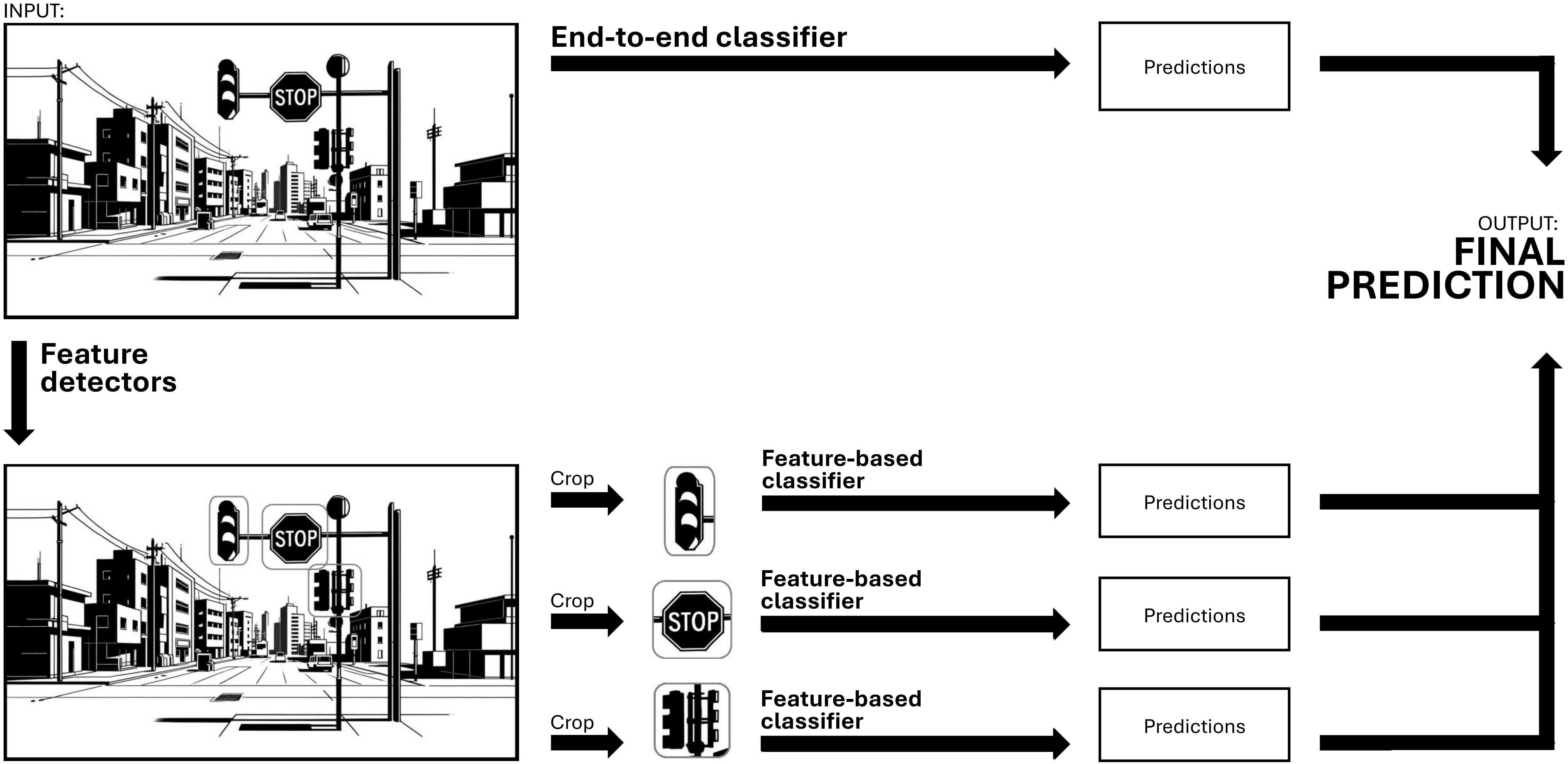
17 cities
Total number of images = **324697**
Total number of places = **57618**



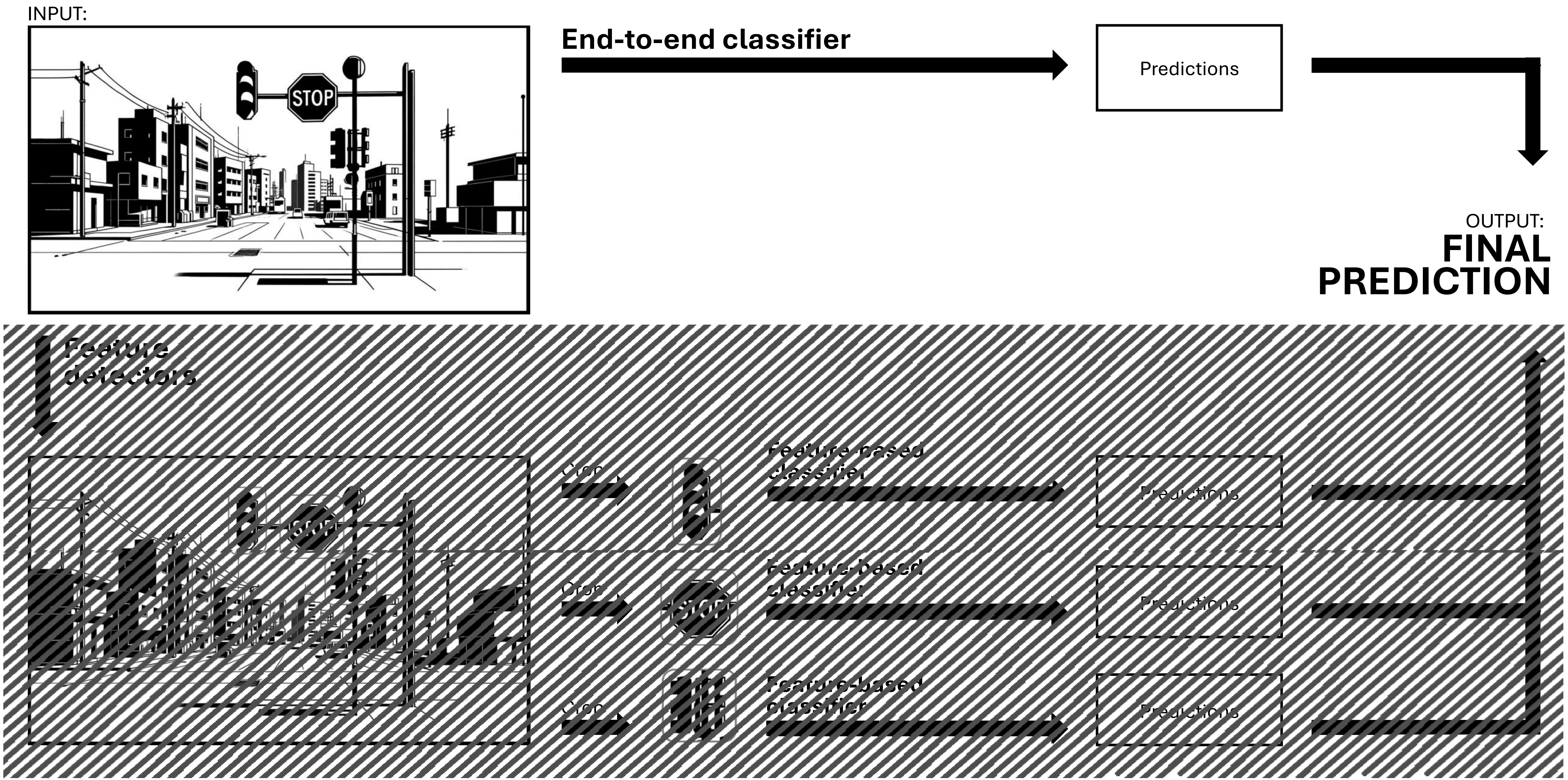


THE WORKFLOW

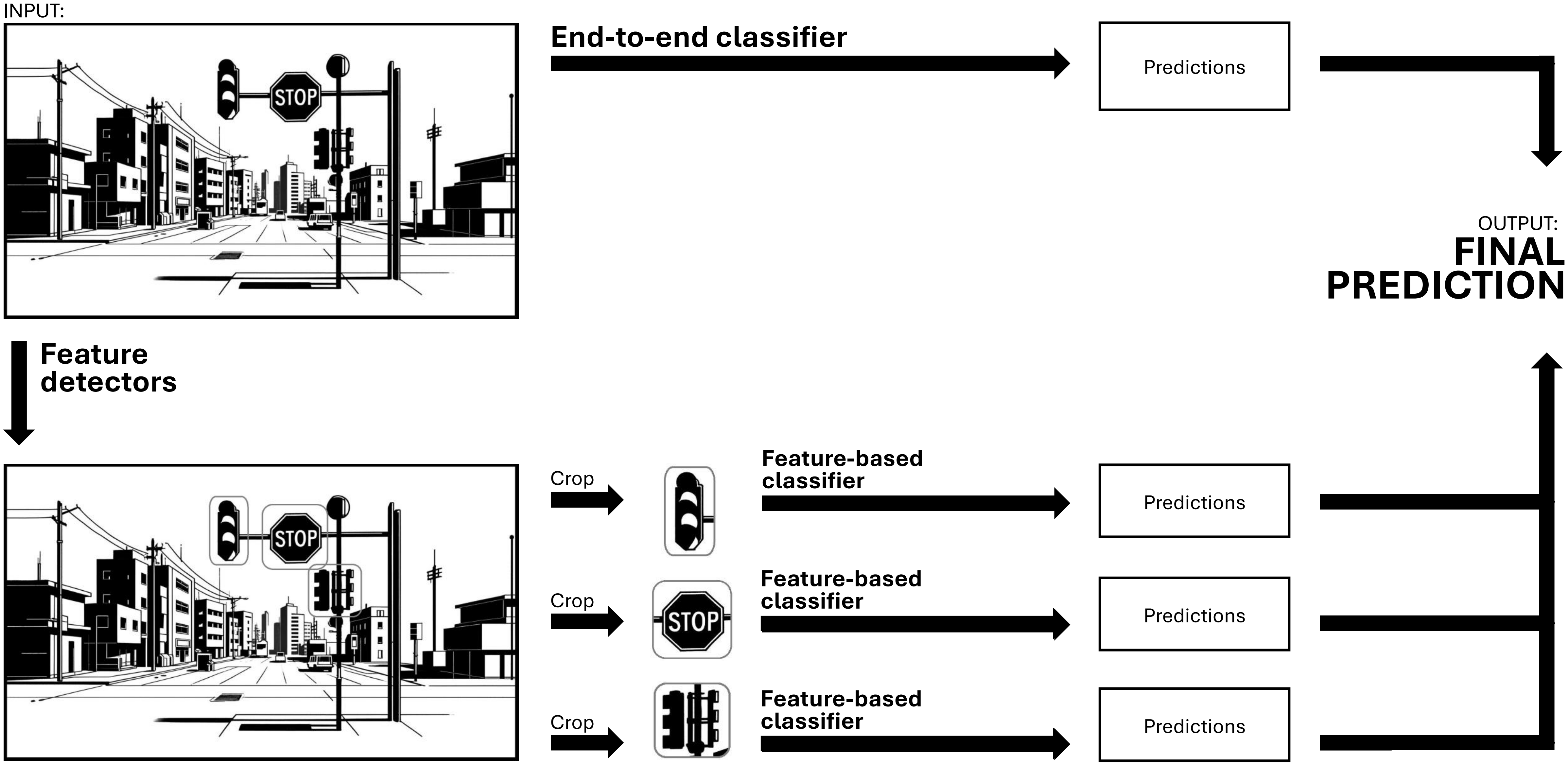
WORKFLOW


INPUT:
End-to-end classifier
Predictions
OUTPUT:
FINAL PREDICTION
Feature detectors
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
STOP

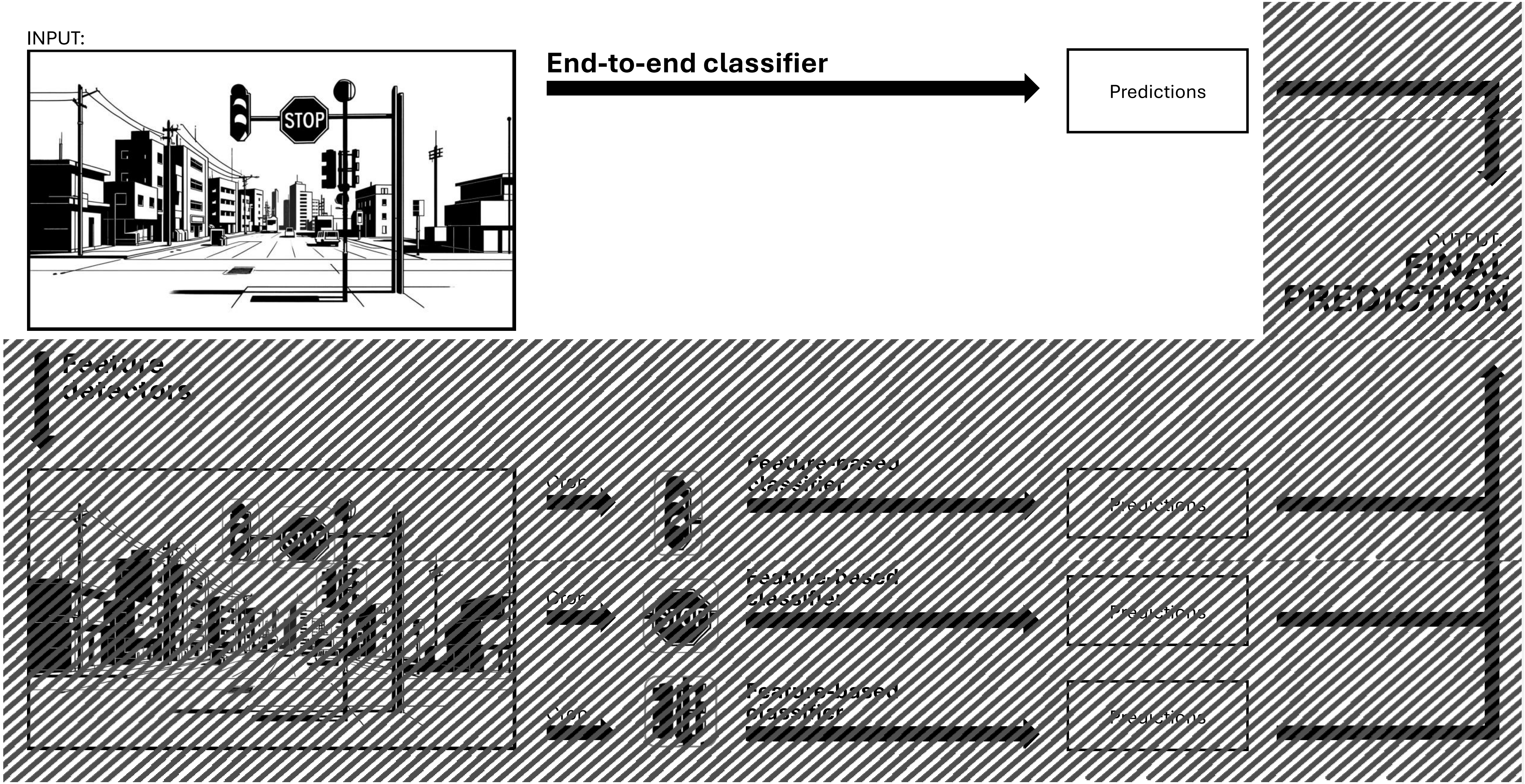
WORKFLOW (NO FEATURES DETECTED)



THE PIPELINE


INPUT:
STOP
End-to-end classifier
Predictions
OUTPUT:
FINAL PREDICTION
Feature detectors
STOP
Crop
Feature-based classifier
Predictions
Crop
STOP
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions

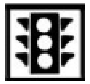
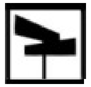
END-TO-END CLASSIFIER



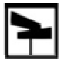
END-TO-END CLASSIFIER

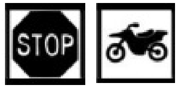
- The architecture of the end-to-end model uses a pre-trained **mobilenetV2** as a backbone, with the addition of **3 dense layers** at the end, which were trained using our balanced dataset.

- For regularisation purposes, the 3 dense layers at the end have **dropout layers** in-between them, which randomly deactivate a certain proportion of neurons of the dense layers for each training iteration, to prevent overfitting.

	BASELINES				PERFORMANCES (ACCURACY)		
Feature	**Baseline (top k)**	**Top 1**	**Top 2**	**Top 3**	**Top 1**	**Top 2**	**Top 3**
$\varnothing$	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r}) \}$	0.094	0.185	0.271	0.634	0.789	0.865
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.169	0.333	0.484	0.595	0.762	0.866
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.263	0.391	0.495	0.536	0.643	0.821
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.325	0.486	0.601	0.750	0.851	0.895
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.128	0.253	0.360	0.636	0.802	0.877
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.349	0.425	0.494	0.632	0.743	0.827
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P}(\mathrm{CITY}_{i_r} \mid$  $) \}$	0.142	0.241	0.321	0.630	0.783	0.865

- Intuitively, if the predictive accuracy of the end-to-end classifier is uncorrelated to whether we detect features or not in the image, then we would expect that the accuracies on the feature-specific domains should be roughly the same as the accuracy on the whole (non-feature-specific) domain. That is, we would expect that the accuracy should stay the same, independently of whether a feature was detected or not. This seems to be the case for the features 

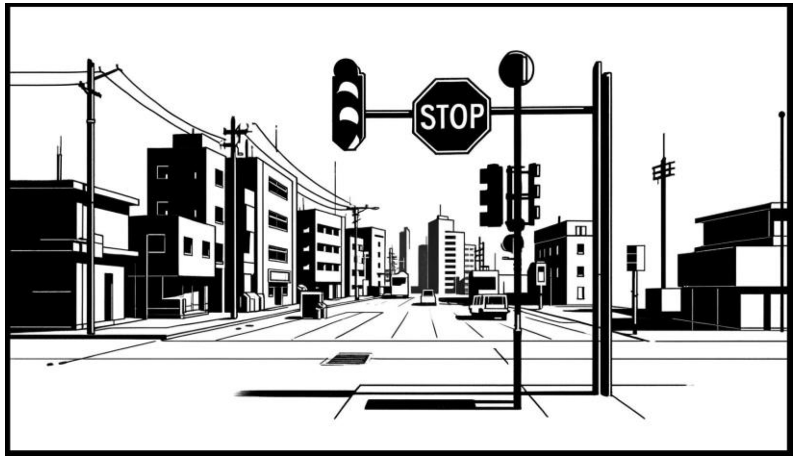
	PERFORMANCES (ACCURACY)		
Feature	Top 1	Top 2	Top 3
∅	0.634	0.789	0.865
	0.595	0.762	0.866
	0.536	0.643	0.821
	0.750	0.851	0.895
	0.636	0.802	0.877
	0.632	0.743	0.827
	0.630	0.783	0.865

- But the accuracies on the feature-specific domains are very different than expected for  !

THIS SUGGESTS THAT, PERHAPS, THE END-TO-END MODEL'S PREDICTION IS SOMEHOW AFFECTED BY WHETHER OR NOT WE DETECTED A MOTORCYCLE (positively affected) OR A STOP SIGN (negatively affected).

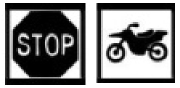
Of course, we can't be completely sure of how the model is affected, because of the lack of transparency in how the model learns to classify images and because there could be other confounding factors.

INPUT:

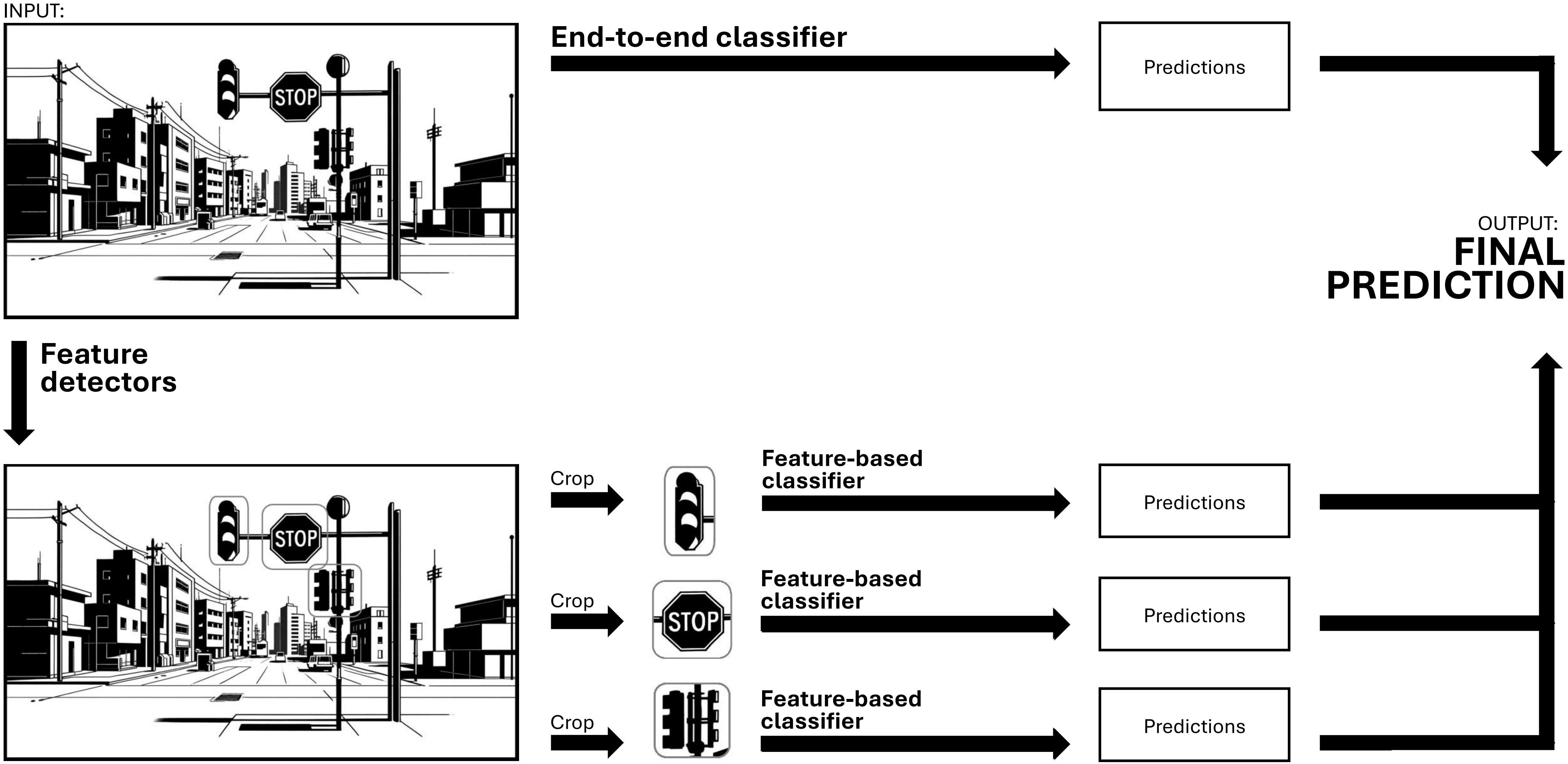


End-to-end classifier

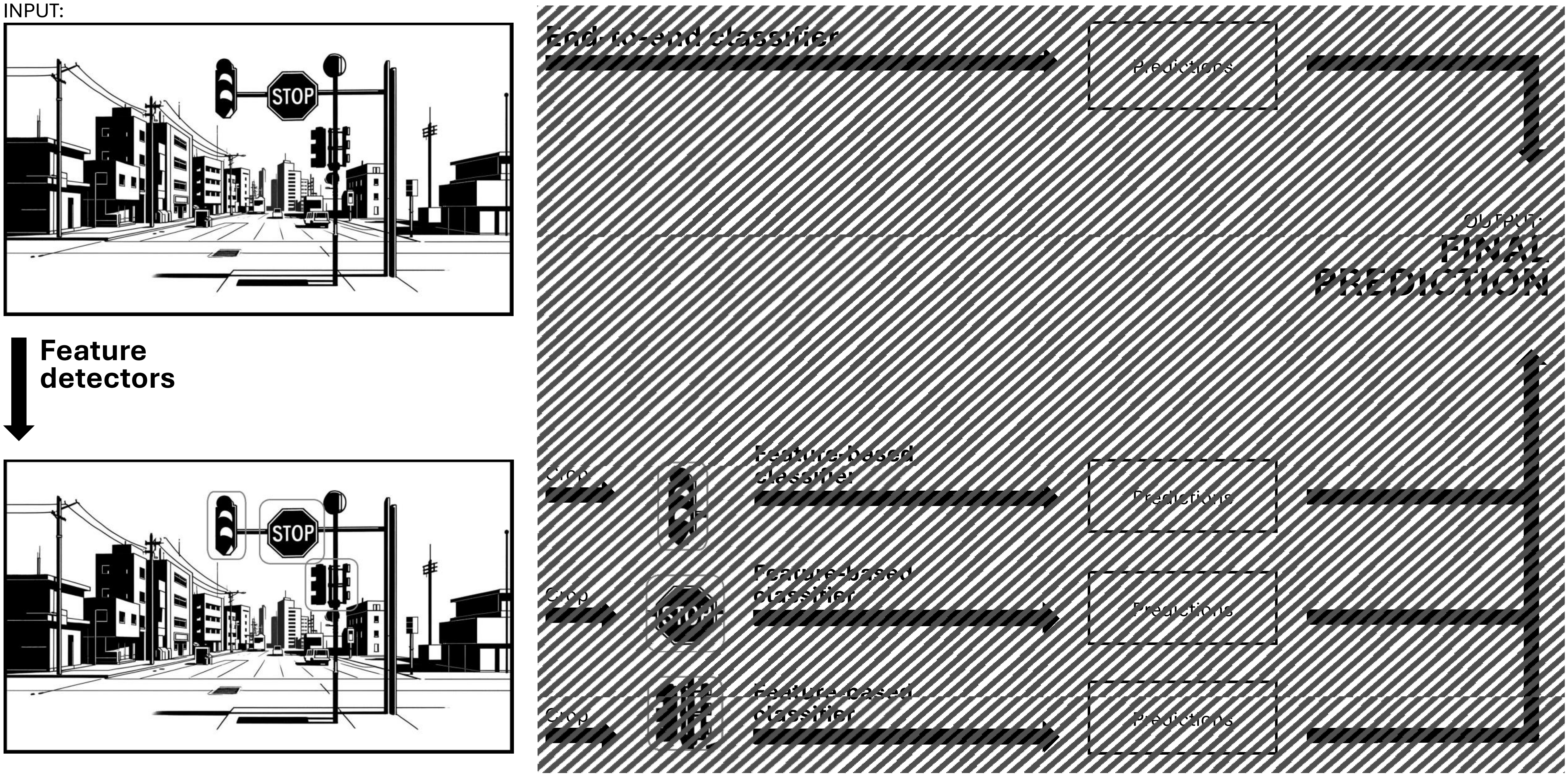
Predictions



THE MODEL MIGHT BE AFFECTED BY THESE FEATURES APPEARING OR NOT


INPUT:
End-to-end classifier
Predictions
Feature detectors
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
OUTPUT:
FINAL PREDICTION

FEATURE DETECTORS

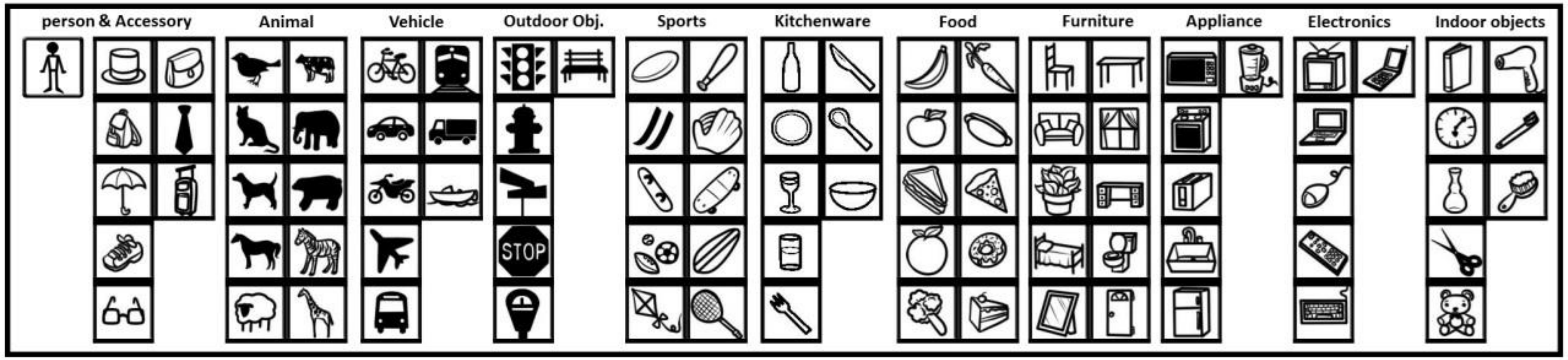


■ We used a neural network model (**ssd_mobilenet_v1_coco_11_06_2017**) pre-trained on

COCO

COMMON **O**BJECTS IN **CO**NTEXT

arxiv:1405.0312

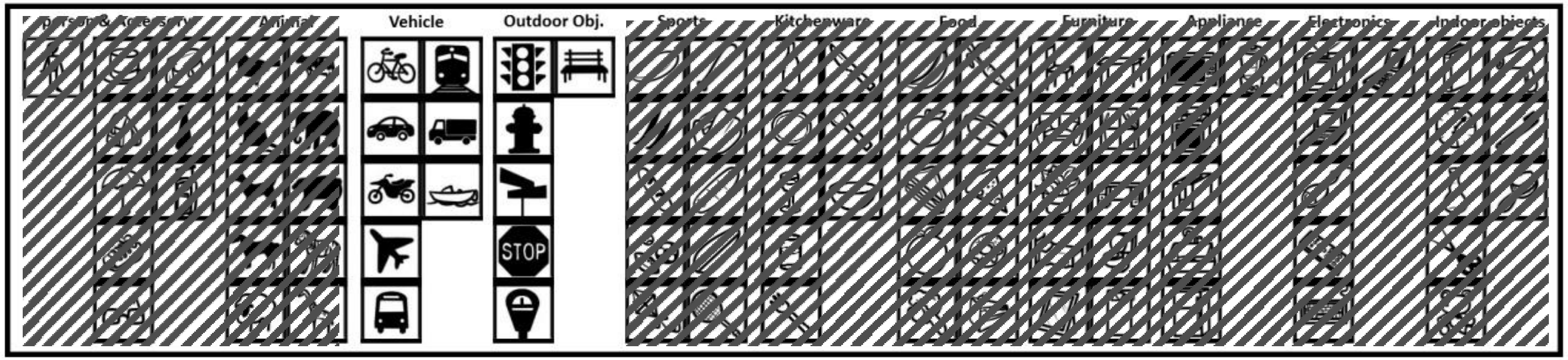


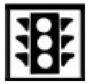
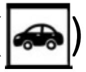
■ We used a neural network model (**ssd_mobilenet_v1_coco_11_06_2017**) pre-trained on

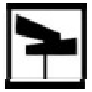
COCO

COMMON **O**BJECTS IN **CO**NTEXT

arxiv:1405.0312

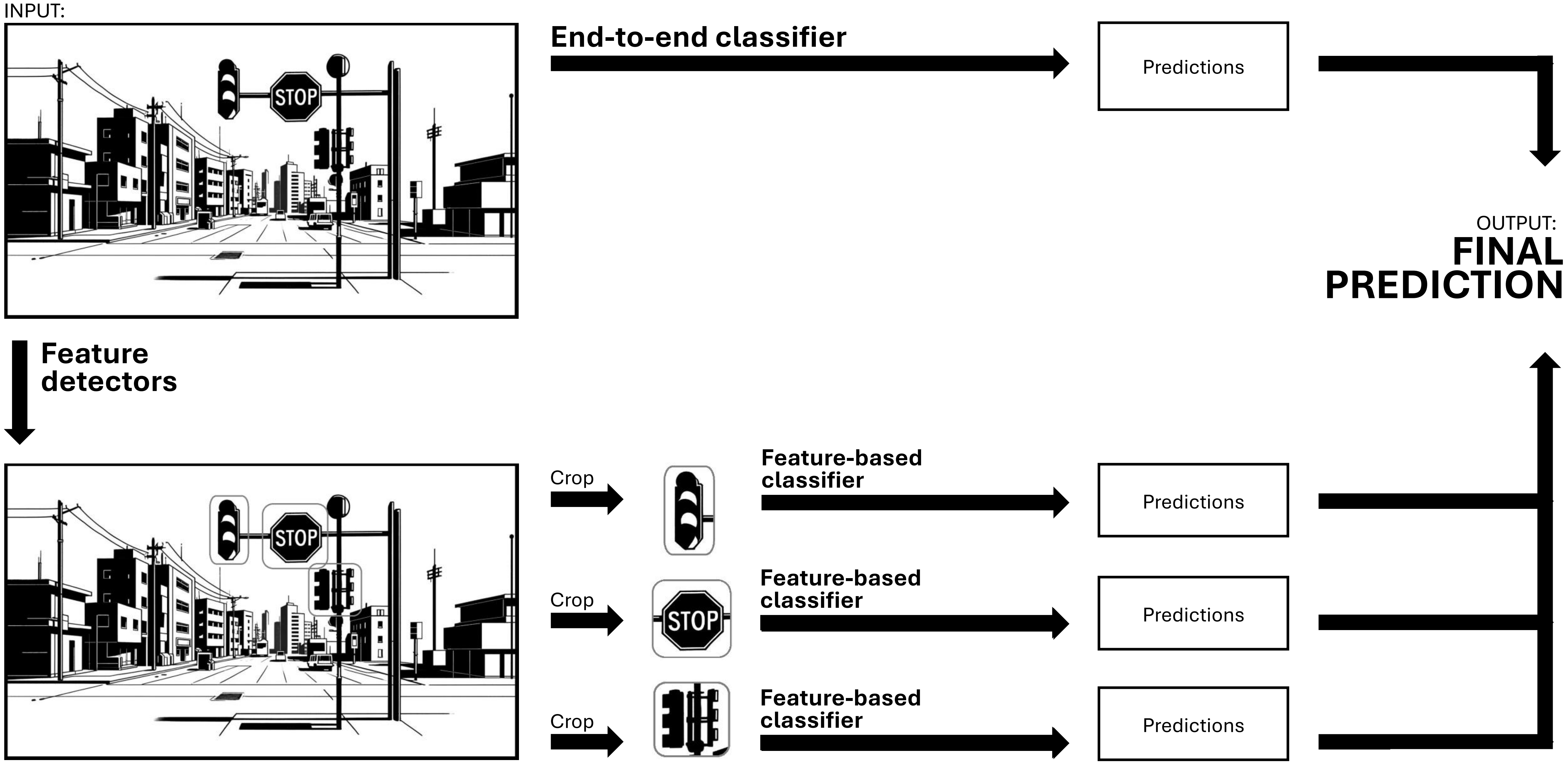


FEATURE	FREQUENCY	TOP 3 CITIES BY FREQUENCY		
	**P**() ≈ 0.01	(1) CHICAGO	(2) LONDON	(3) PHOENIX
	**P**() ≈ 0.0025	(1) MIAMI	(2) CHICAGO	(3) BOSTON
	**P**() ≈ 0.005	(1) BANGKOK	(2) ROME	(3) LONDON
	**P**() ≈ 0.418*	(1) LONDON	(2) LISBON	(3) ROME
	**P**() ≈ 0.009	(1) LONDON	(2) ROME	(3) PRS

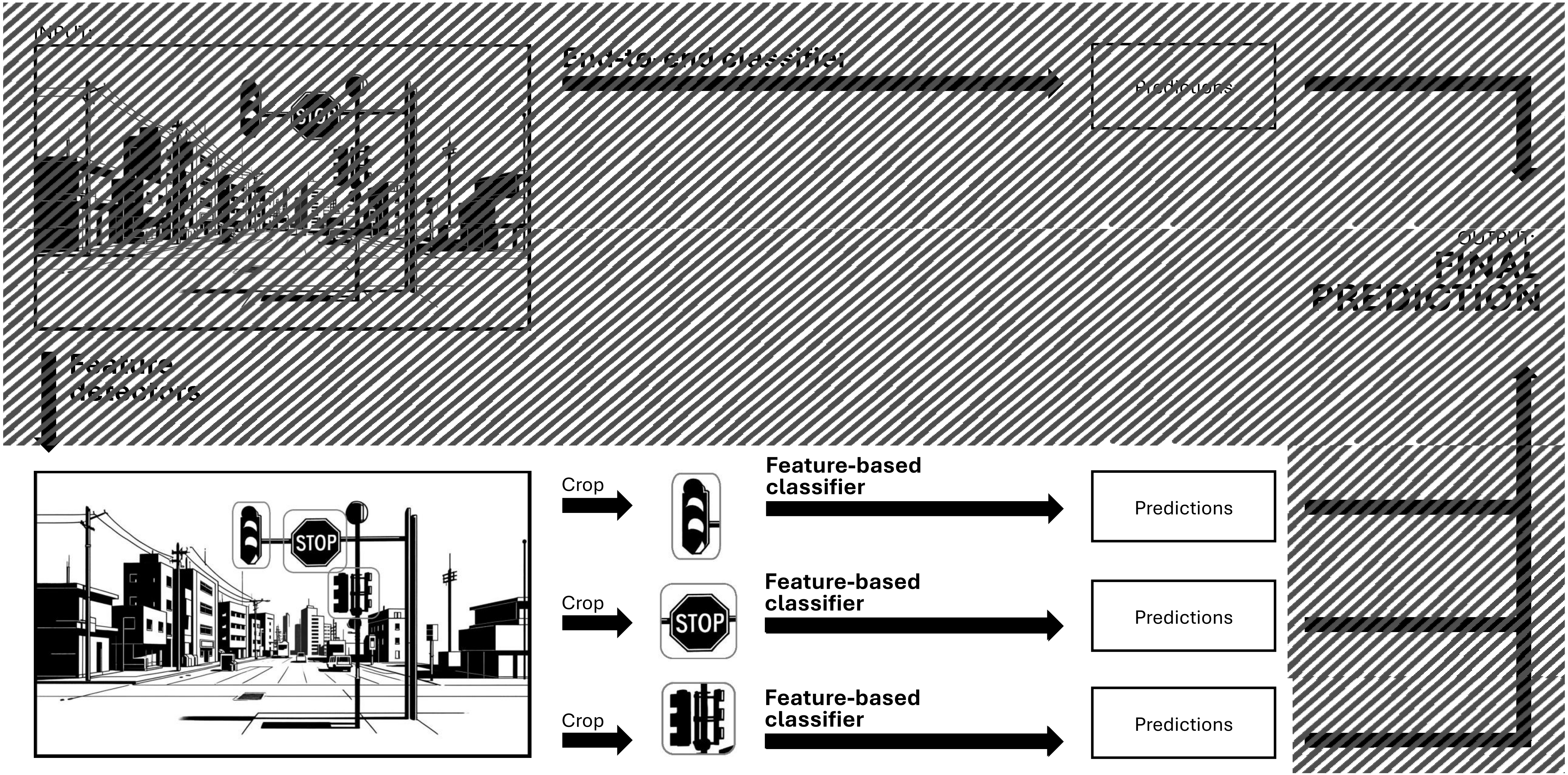


- We also used a Faster RCNN with a pretrained ResNet50 backbone, fine-tuned for detecting traffic signs with the **German Traffic Sign Recognition Benchmark** (**GTSRB**) dataset. This detector, however, was much less reliable than COCO, and generated a very noisy dataset of cropped images for the feature-based classifier.

- Due to the high levels of detection noise, it is difficult to estimate the frequency of the “traffic signs” feature.


INPUT:
End-to-end classifier
Predictions
OUTPUT:
FINAL PREDICTION
Feature detectors
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
STOP

FEATURE-BASED CLASSIFIERS

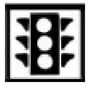
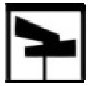


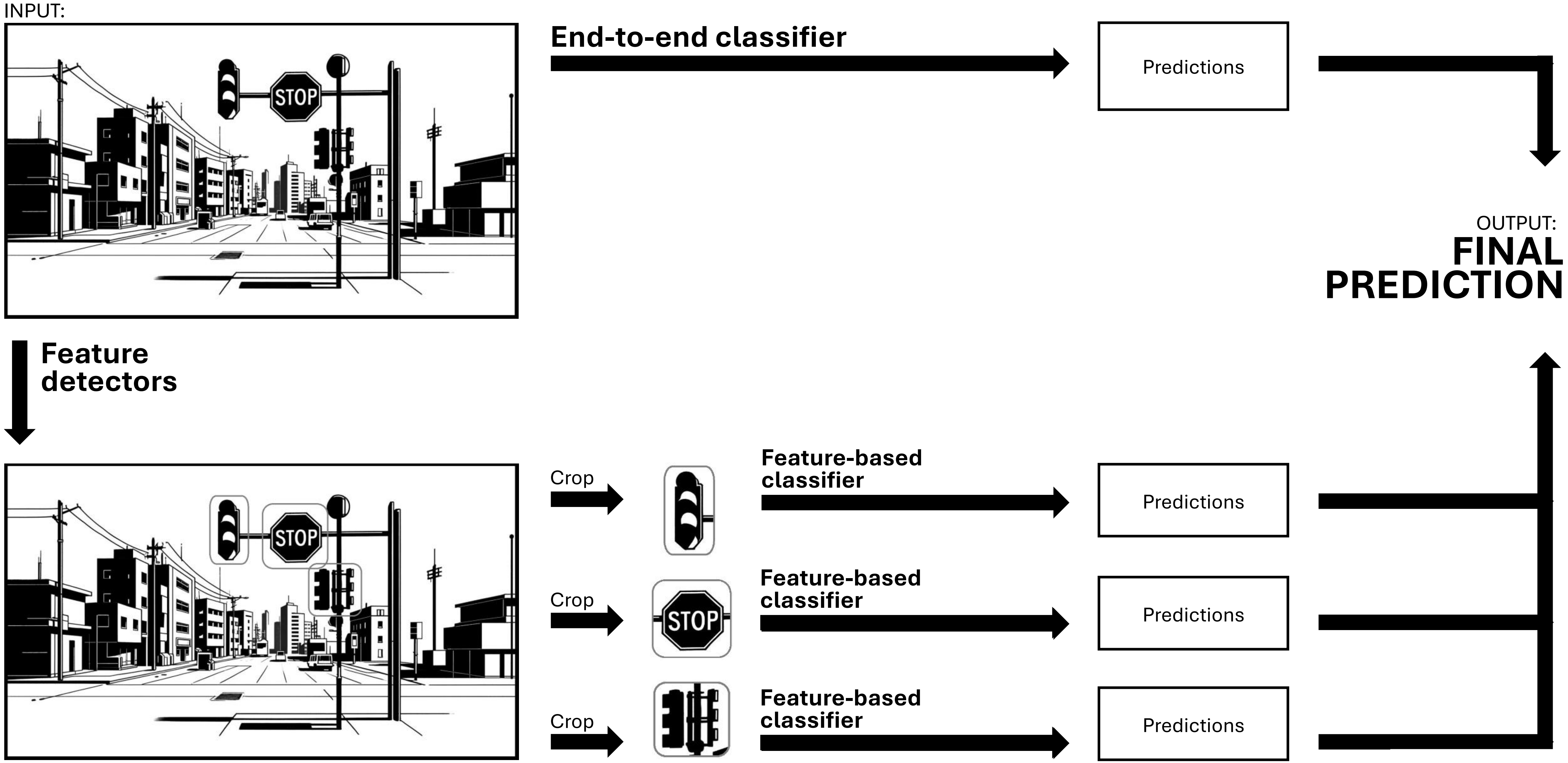
FEATURE-BASED CLASSIFIERS

- Basic CNN model:

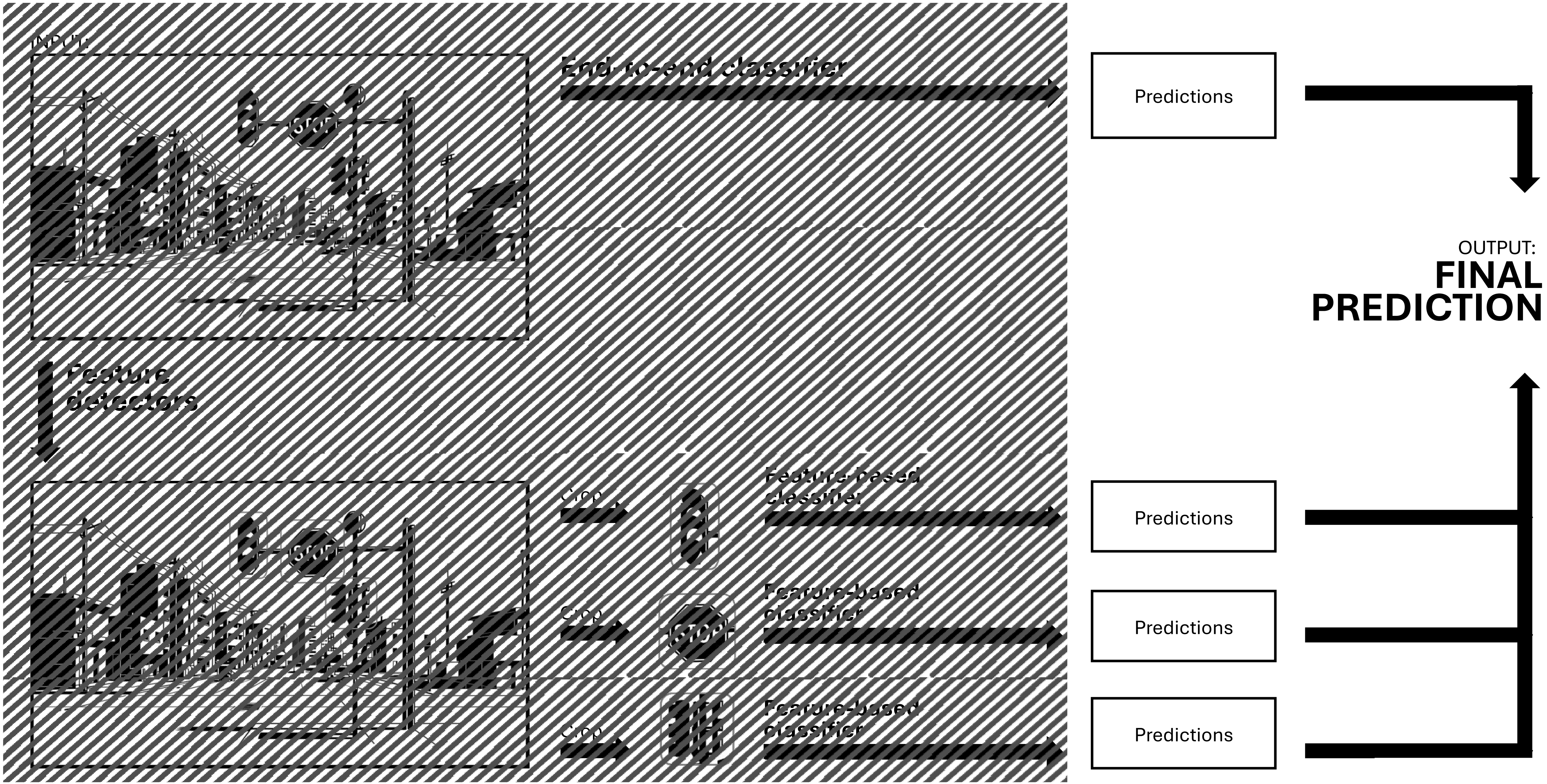
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_1 (Rescaling)     (None, 50, 50, 3)         0

 conv2d_2 (Conv2D)           (None, 50, 50, 16)        448

 max_pooling2d_2 (MaxPooling  (None, 25, 25, 16)       0
 2D)

 conv2d_3 (Conv2D)           (None, 25, 25, 32)        4640

 max_pooling2d_3 (MaxPooling  (None, 12, 12, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 4608)              0

 dense_2 (Dense)             (None, 64)                294976

 dense_3 (Dense)             (None, 23)                1495

=================================================================
Total params: 301,559
Trainable params: 301,559
Non-trainable params: 0
```

- Trained on the datasets created by using the feature detectors to crop images of the specified features from the images in the balanced dataset
- Not enough quality training data right now to be able to satisfactorily train more complex model architectures

	BASELINES				PERFORMANCES (ACCURACY)
Feature	**Baseline (top k)**	**Top 1**	**Top 2**	**Top 3**	**Top 1**
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.169	0.333	0.484	0.264
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.263	0.391	0.495	0.363
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.325	0.486	0.601	0.444
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.128	0.253	0.360	0.208
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.349	0.425	0.494	0.416
	$\max_{i_1<\ldots<i_k} \{ \sum_{r=1}^{k} \mathbb{P} (\mathrm{CITY}_{i_r} \mid$  $) \}$	0.142	0.241	0.321	0.208


INPUT:
STOP
End-to-end classifier
Predictions
OUTPUT:
FINAL PREDICTION
Feature detectors
STOP
Crop
Feature-based classifier
Predictions
Crop
STOP
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions

ENSEMBLE



INPUT:
End-to-end classifier
Predictions
OUTPUT:
FINAL PREDICTION
Feature detectors
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions
Crop
Feature-based classifier
Predictions

ENSEMBLE

- In order to aggregate and combine together the prediction coming from the end-to-end model and the predictions coming from the feature-based classifiers (if available), we weight the sum of the predictions with a weighting scheme that also takes into account the confidence levels of the various object detectors in detecting features (if available).

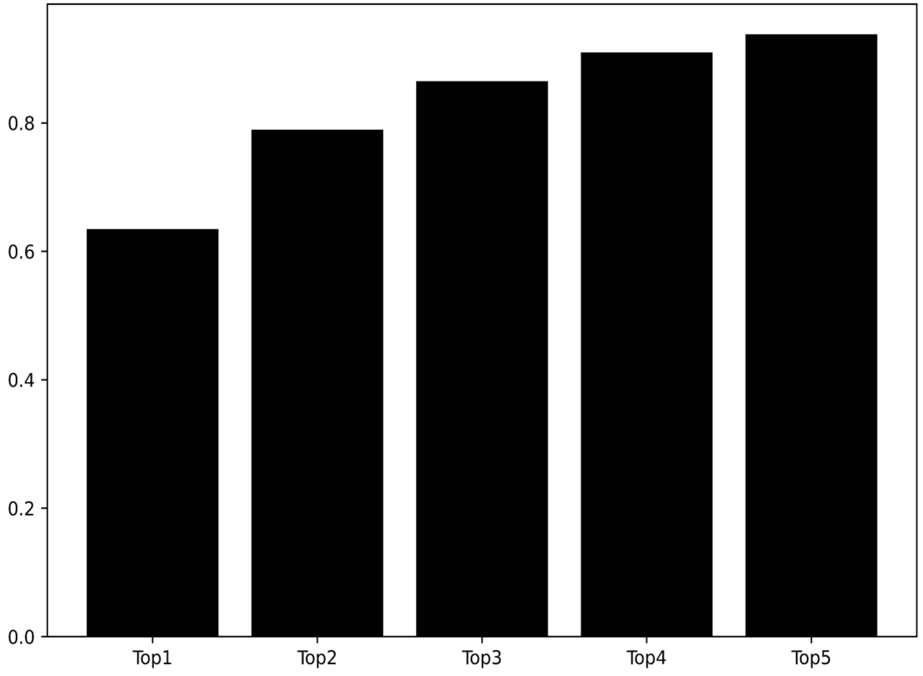
- At the moment, the weighting scheme largely favours the end-to-end model’s prediction, because of the current better performances of the end-to-end model compared to that of the feature-based classifiers on the feature-specific domains, but this could change if one manages to get better feature-based classifiers.

RESULTS

PERFORMANCE ANALYSIS

- Performance results of the complete pipeline according to various metrics

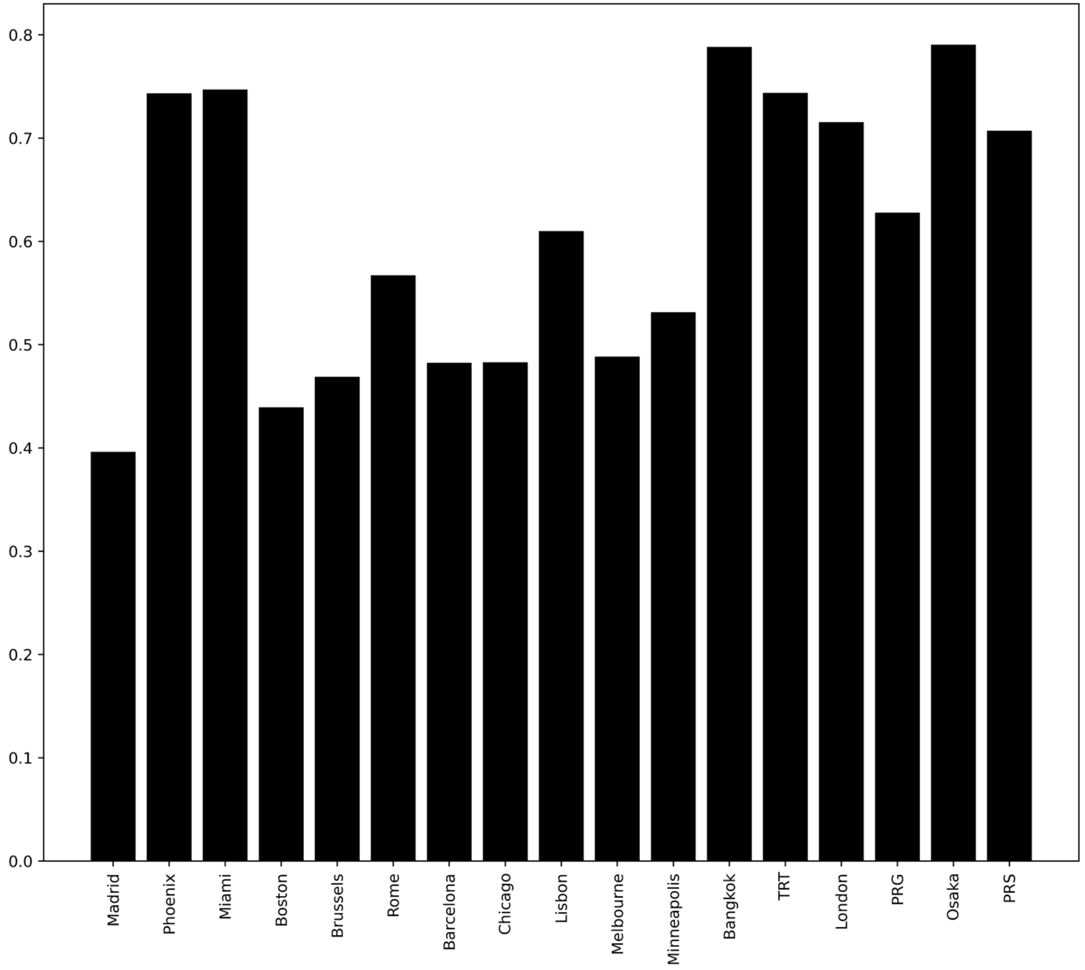
ACCURACY



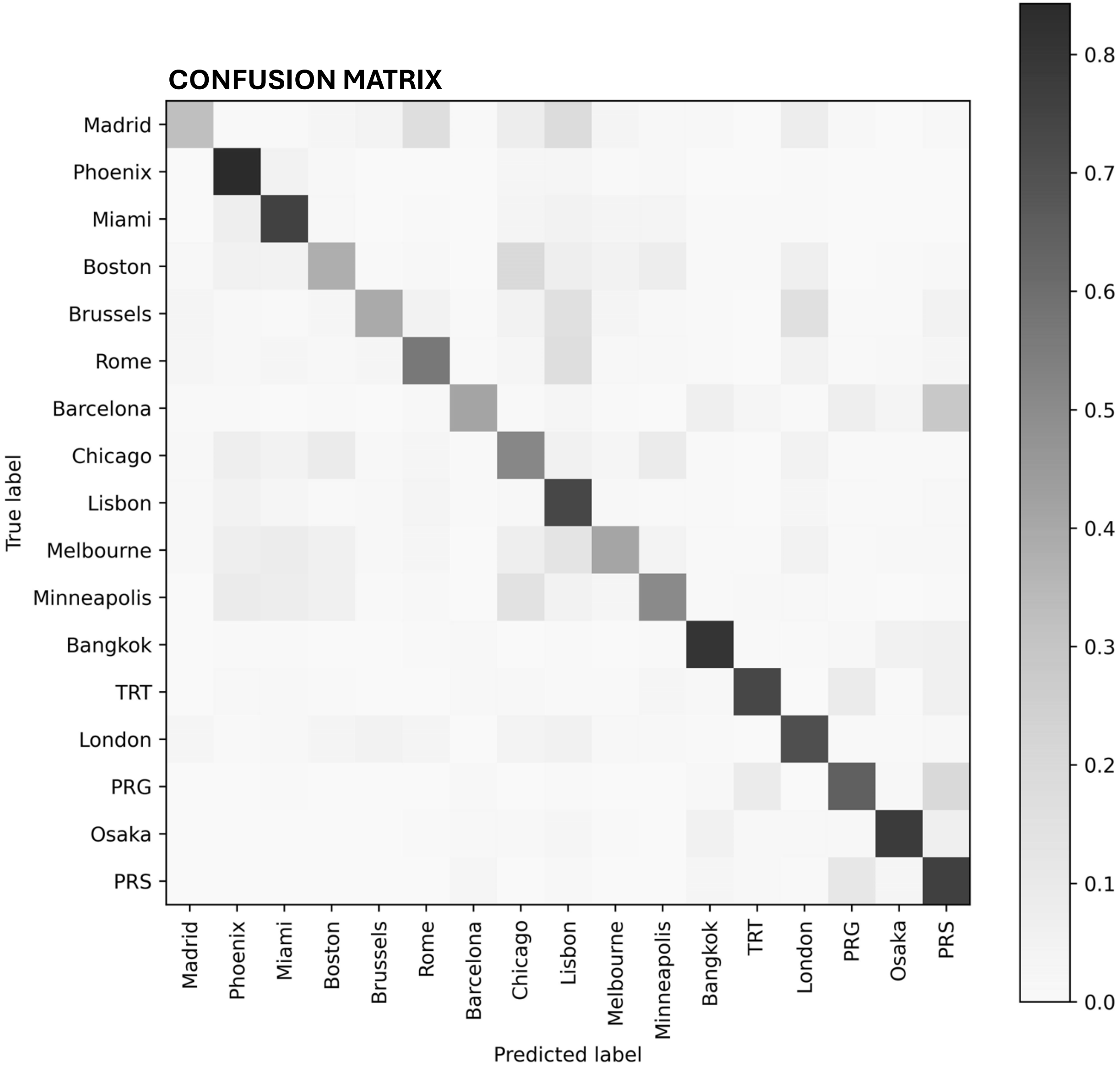
FINAL ACCURACY: 0.635

(essentially the end-to-end model)

F1 SCORES



- The model seems to mix up cities from similar geographic areas, but is able to distinguish between different geographic regions fairly well!



FUTURE IMPROVEMENTS

- Improve the feature-based classifiers by getting more quality data for the training, so to be able to also explore more complex models.
- Add more features (a starting point could be to add all the “COCO outdoors objects” features).
- Include rural areas and use texture-based features (such as GCLM).
- Improve the end-to-end model by experimenting with other architectures.
- Optimise the final model’s ensemble weighting and explore other ways to aggregate and combine the predictions from the classifiers and the end-to-end model.

THANK YOU

MADRID

ROME

MADRID

ROME

DEMO AVAILABLE AT

https://github.com/hochfilzer/geo-locator