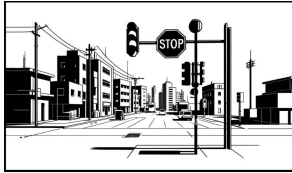


# GEO-LOCATOR



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



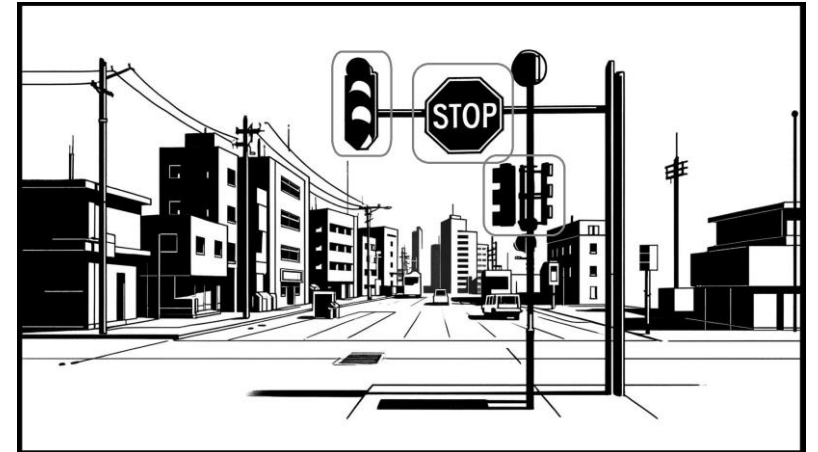
ROME 



## 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:

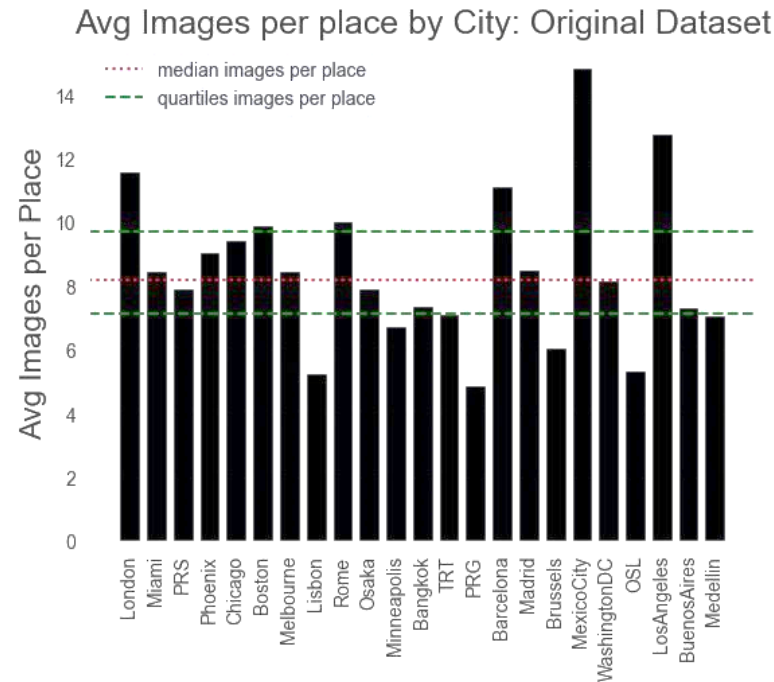
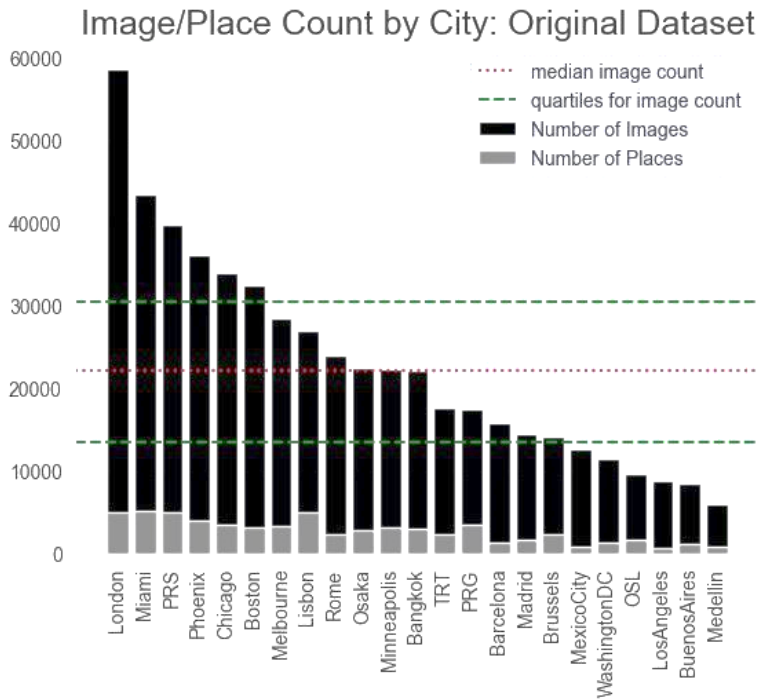
	place_id	year	month	northdeg	city_id	lat	lon	panoid
0	1678	2014	10	370	Barcelona	41.402066	2.198988	DB4DzIzCRq4IyE9FMx_9Ow

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

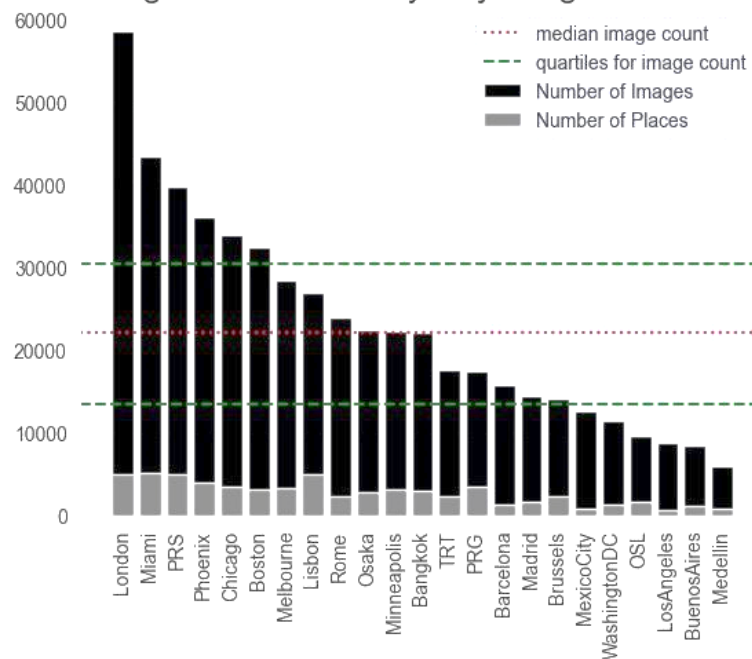
# BALANCED DATASET

17 cities

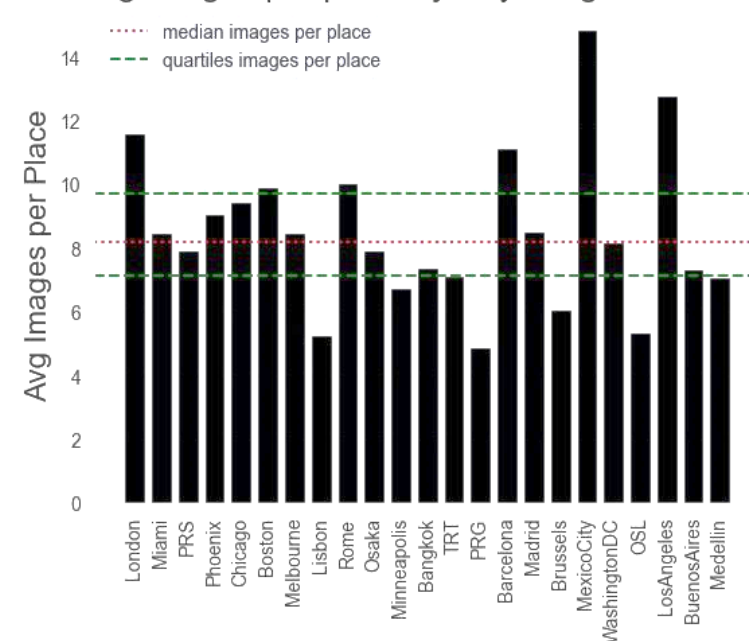
Total number of images = **324697**

Total number of places = **57618**

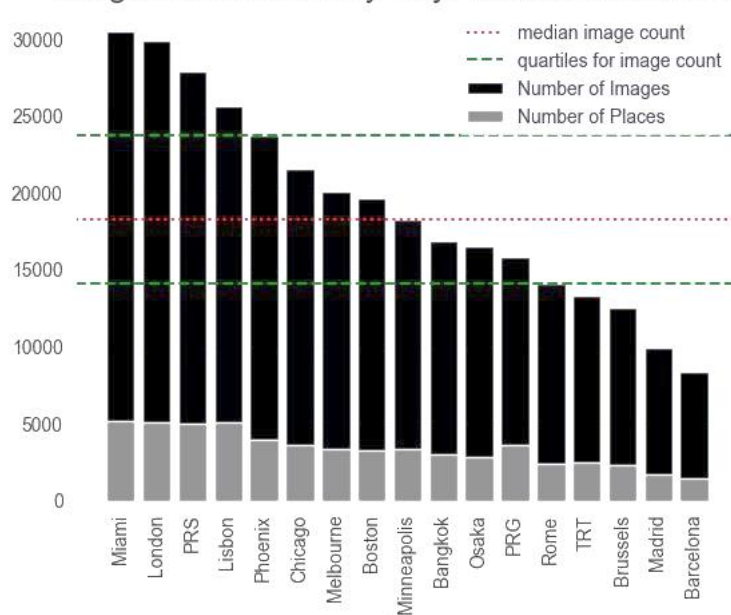
Image/Place Count by City: Original Dataset



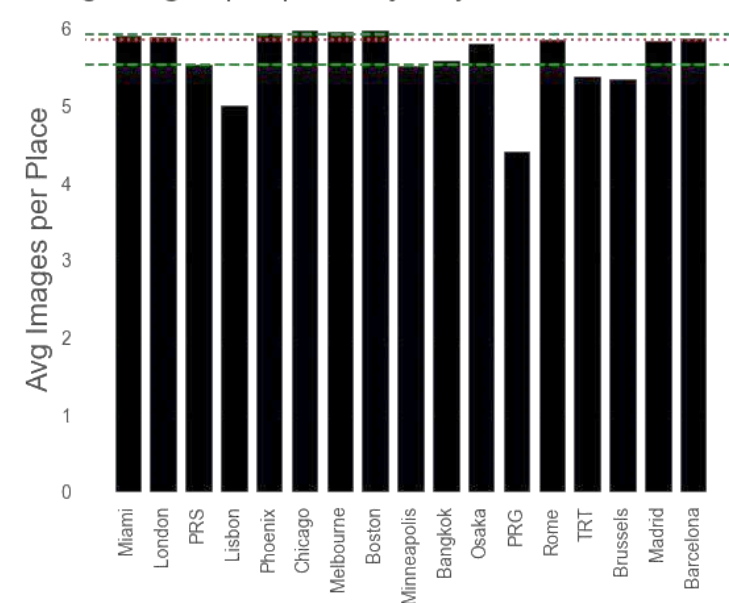
Avg Images per place by City: Original Dataset



Image/Place Count by City: Normalized Dataset



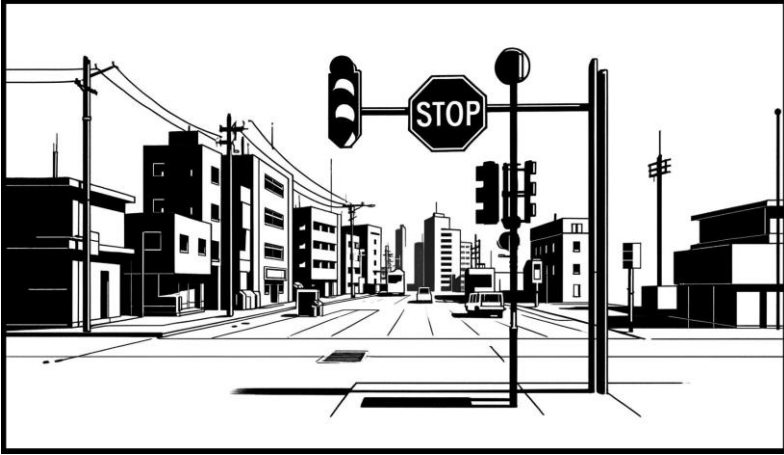
Avg Images per place by City: Normalized Dataset



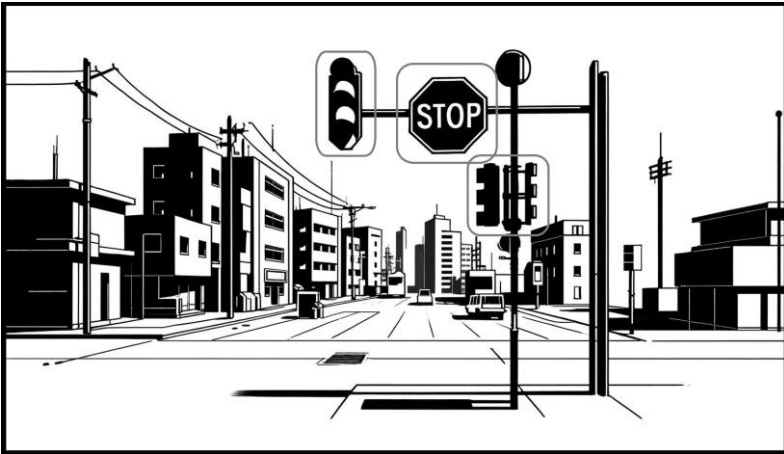
# **THE WORKFLOW**

# WORKFLOW

INPUT:



Feature  
detectors



End-to-end classifier



Predictions



OUTPUT:

**FINAL  
PREDICTION**

Crop



Feature-based  
classifier



Predictions

Crop



Feature-based  
classifier



Predictions

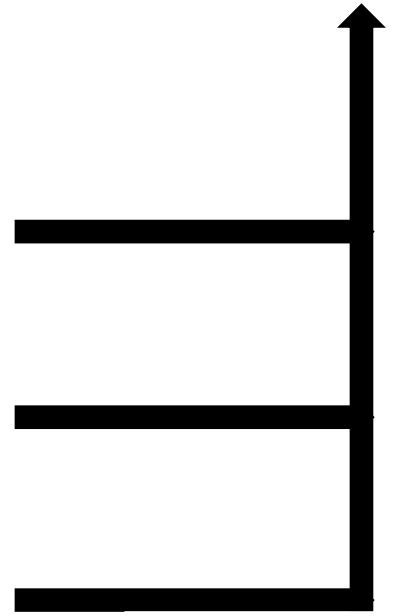
Crop



Feature-based  
classifier



Predictions



# WORKFLOW ( NO FEATURES DETECTED )

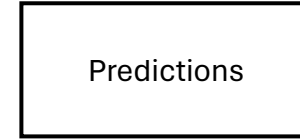
INPUT:



End-to-end classifier



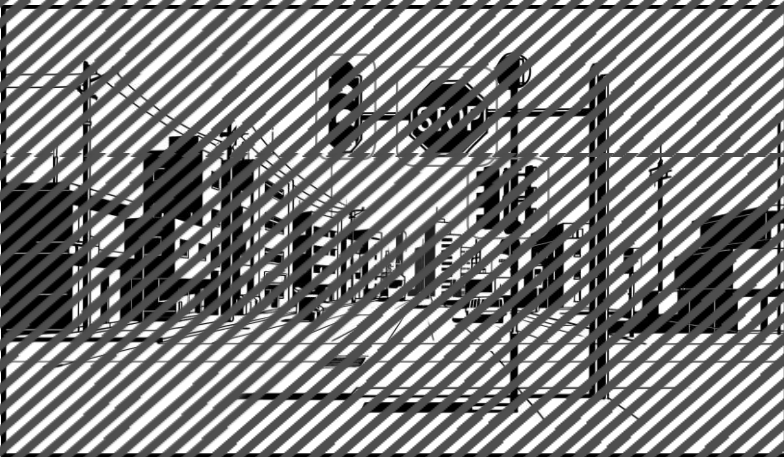
Predictions



OUTPUT:

**FINAL  
PREDICTION**

Feature  
detectors



Feature-based  
classifier



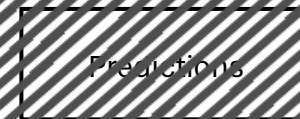
Predictions



Feature-based  
classifier



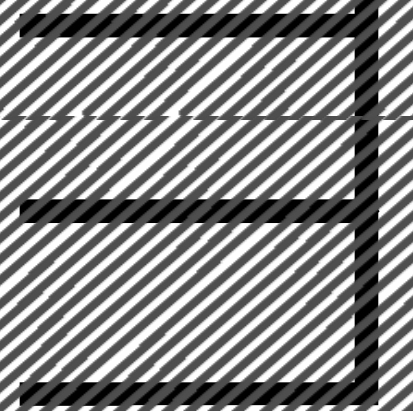
Predictions



Feature-based  
classifier



Predictions



# **THE PIPELINE**



INPUT:



End-to-end classifier



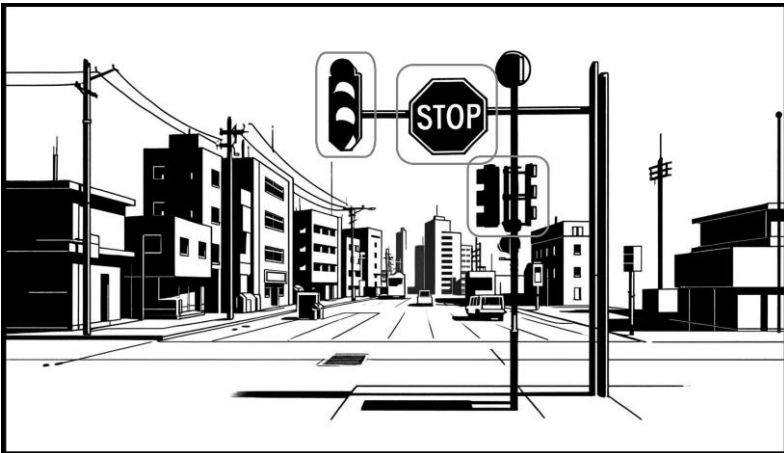
Predictions



OUTPUT:

**FINAL  
PREDICTION**

Feature  
detectors



Crop



Feature-based  
classifier



Predictions

Crop

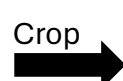


Feature-based  
classifier



Predictions

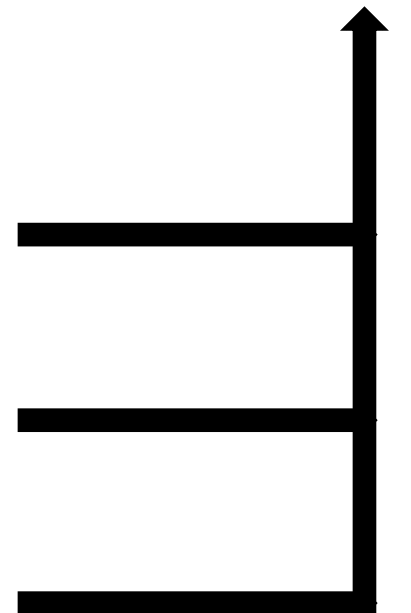
Crop



Feature-based  
classifier

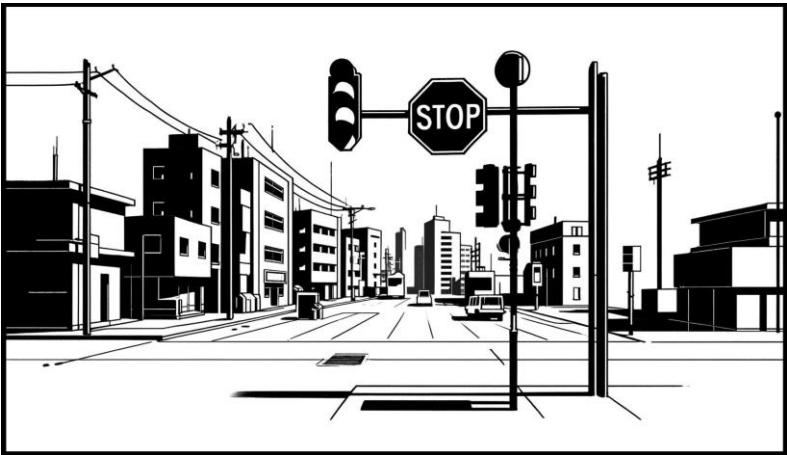


Predictions



# END-TO-END CLASSIFIER

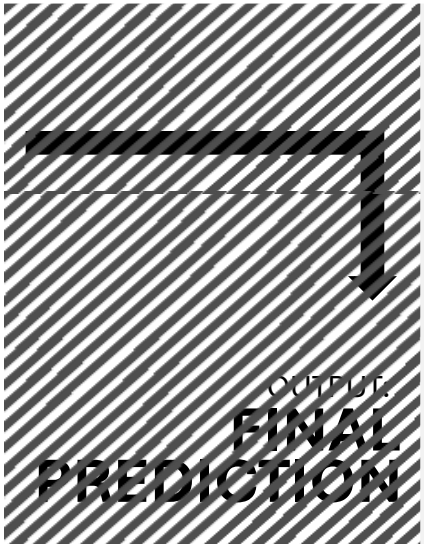
INPUT:



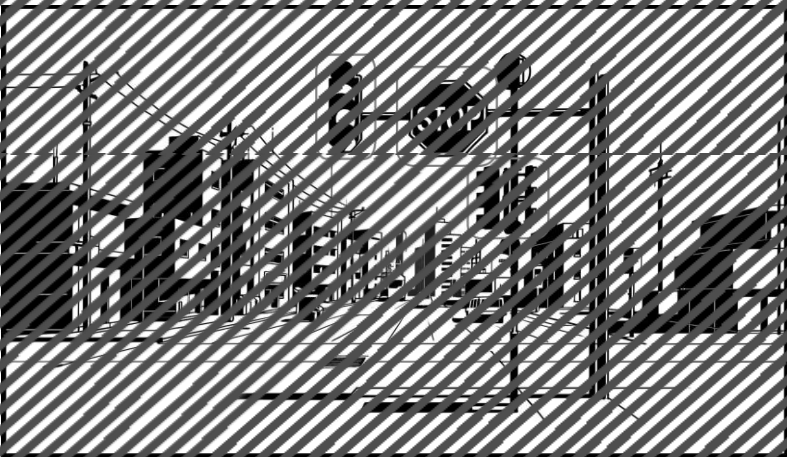
End-to-end classifier



Predictions



Feature detectors



Crop



Feature-based classifier



Predictions

Crop



Feature-based classifier



Predictions

Crop










Feature-based classifier









Predictions

# 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.

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

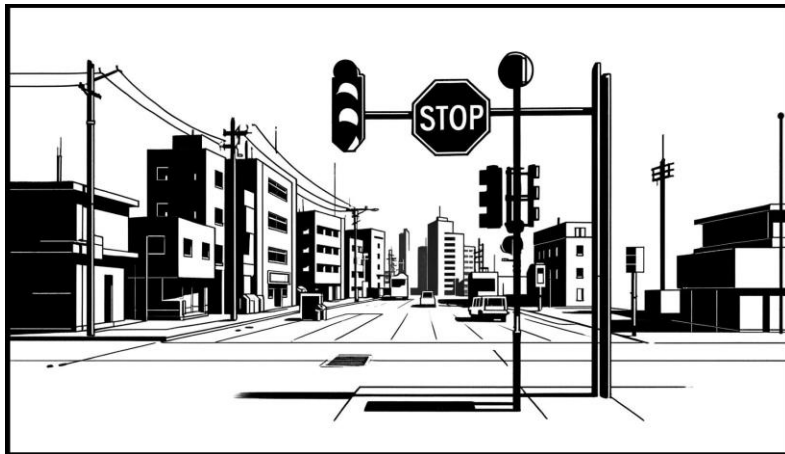
- Intuitively, if the end-to-end classifier does not take into account features in its prediction, then we would expect that the accuracies on the feature-specific domains should be roughly the same as the accuracy on the whole (non-feature-specific) domain. That is, we would expect that the accuracy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

But the accuracies on the feature-specific domains are 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 (+) OR A STOP SIGN (-).**

(Of course, we can't be completely sure of this because, since lack of transparency in how the model learns to classify means that there could be confounding factors.)

INPUT:



**End-to-end classifier**



THE MODEL MIGHT TAKE  
SOME OF THE FEATURES  
INTO ACCOUNT !

Predictions



INPUT:



End-to-end classifier



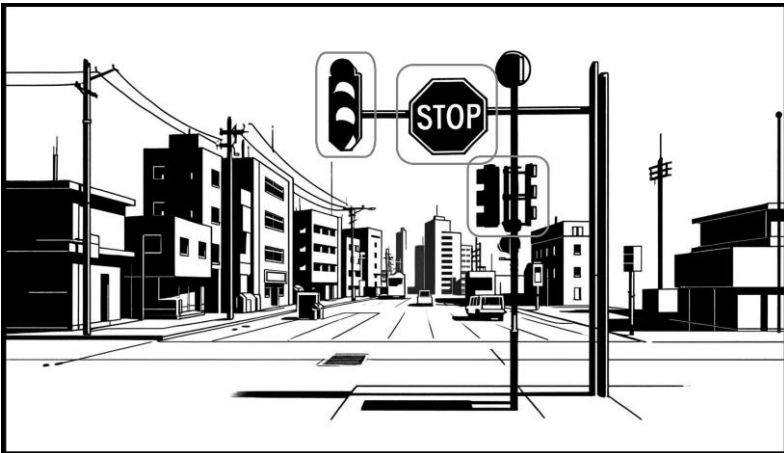
Predictions



OUTPUT:

**FINAL  
PREDICTION**

Feature  
detectors



Crop



Feature-based  
classifier



Predictions

Crop

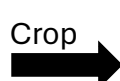


Feature-based  
classifier



Predictions

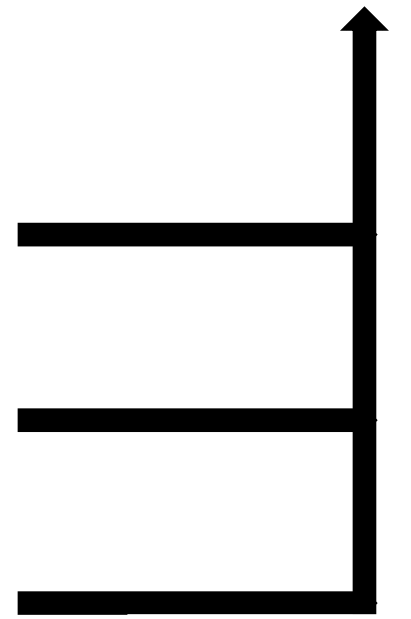
Crop



Feature-based  
classifier

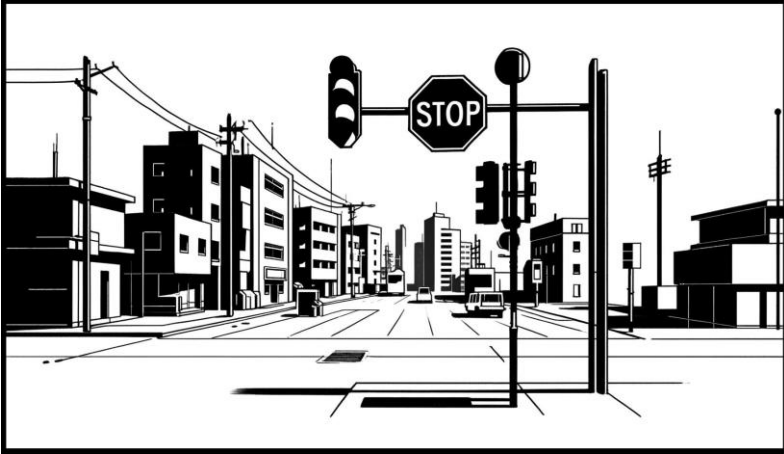


Predictions

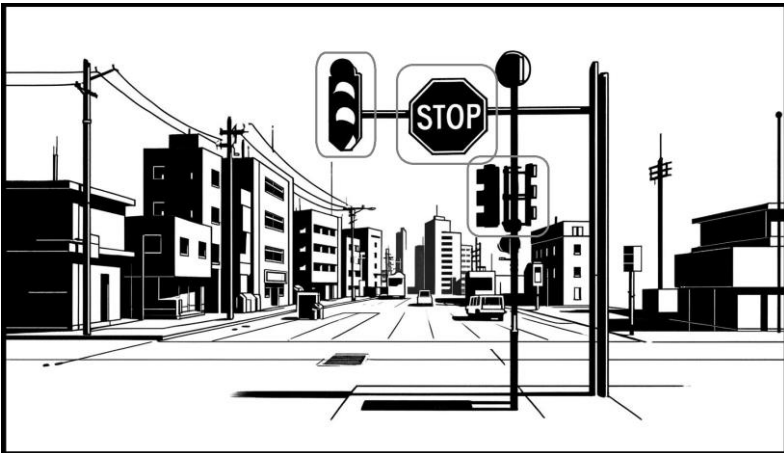


# FEATURE DETECTORS

INPUT:



Feature  
detectors



End-to-end classifier



Predictions



OUTPUT:

**FINAL  
PREDICTION**

Crop



Feature-based  
classifier



Predictions

Crop



Feature-based  
classifier



Predictions

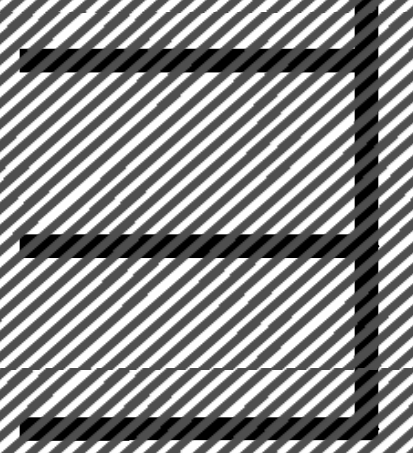
Crop



Feature-based  
classifier



Predictions

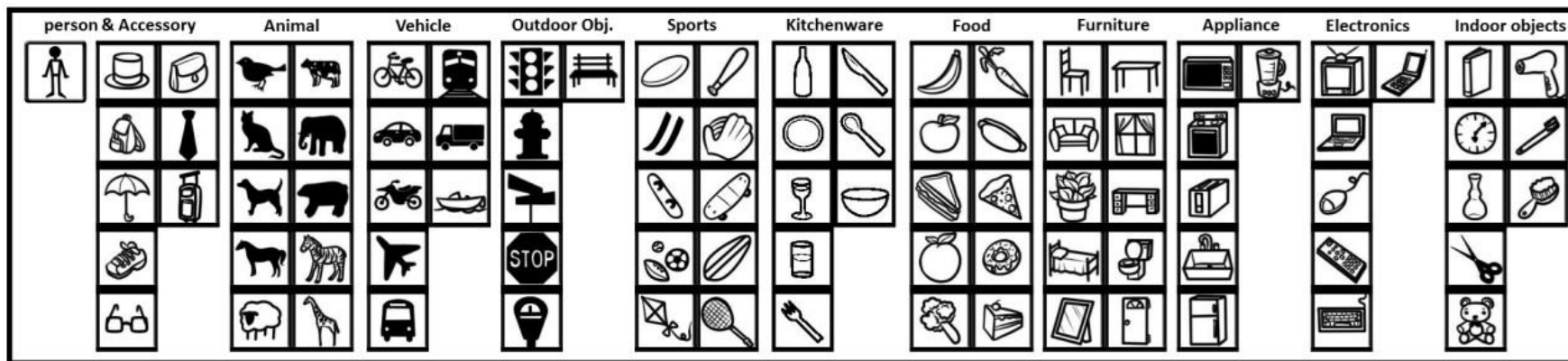


■ We used a neural network model ( `ssd_mobilenet_v1_coco_11_06_2017` ) pre-trained on

# COCO

COMMON OBJECTS IN CONTEXT

arxiv:1405.0312

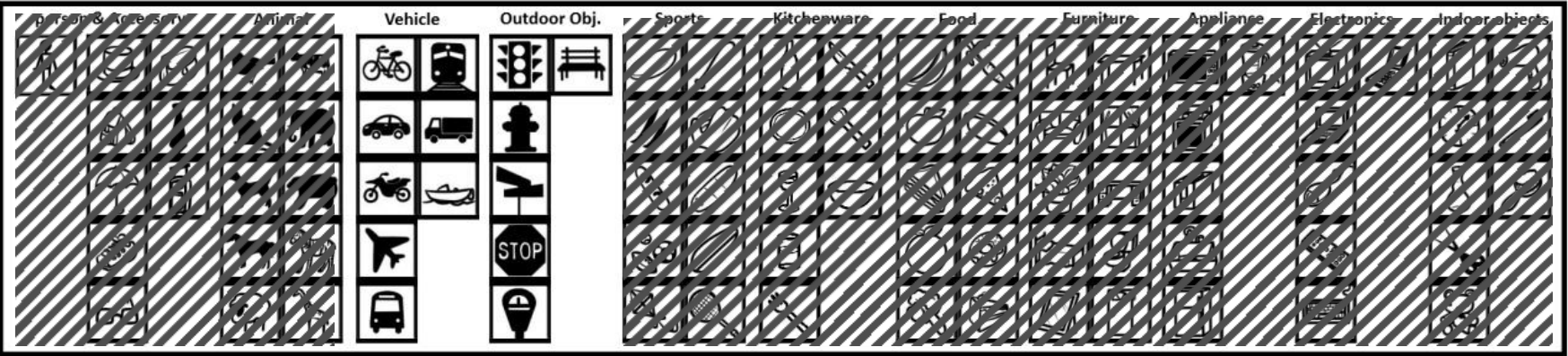


■ We used a neural network model ( `ssd_mobilenet_v1_coco_11_06_2017` ) pre-trained on

# COCO

COMMON OBJECTS IN CONTEXT

arxiv:1405.0312



FEATURE	FREQUENCY	TOP 3 CITIES BY FREQUENCY
	$P(\text{Traffic light}) \approx 0.01$	(1) CHICAGO (2) LONDON (3) PHOENIX
	$P(\text{STOP sign}) \approx 0.0025$	(1) MIAMI (2) CHICAGO (3) BOSTON
	$P(\text{Motorcycle}) \approx 0.005$	(1) BANGKOK (2) ROME (3) LONDON
	$P(\text{Car}) \approx 0.418^*$	(1) LONDON (2) LISBON (3) ROME
	$P(\text{Bus}) \approx 0.009$	(1) LONDON (2) ROME (3) PRS

INPUT:



End-to-end classifier



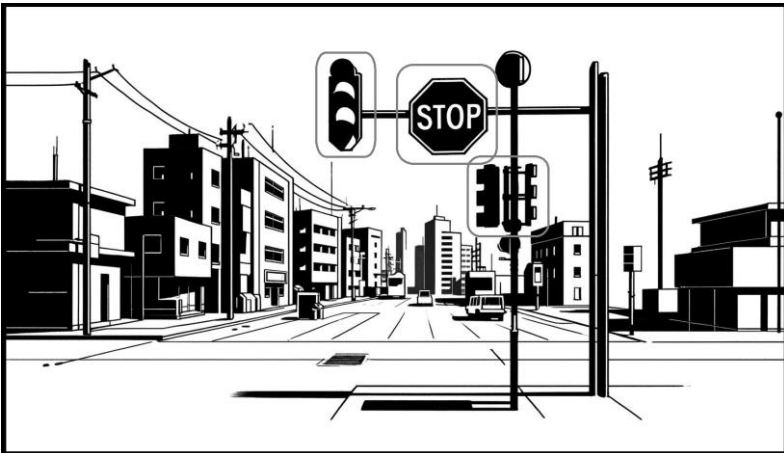
Predictions



OUTPUT:

**FINAL  
PREDICTION**

Feature  
detectors



Crop



Feature-based  
classifier



Predictions

Crop

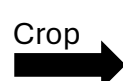


Feature-based  
classifier



Predictions

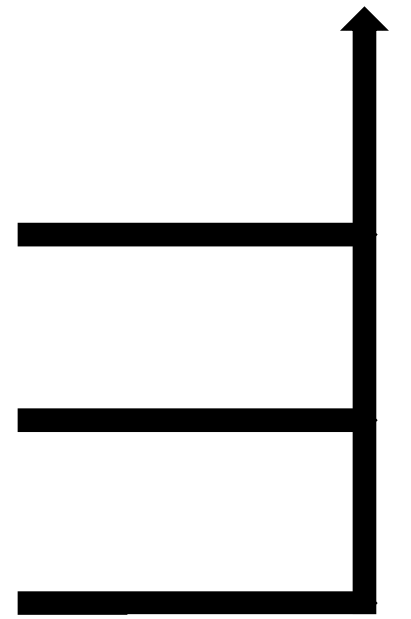
Crop



Feature-based  
classifier

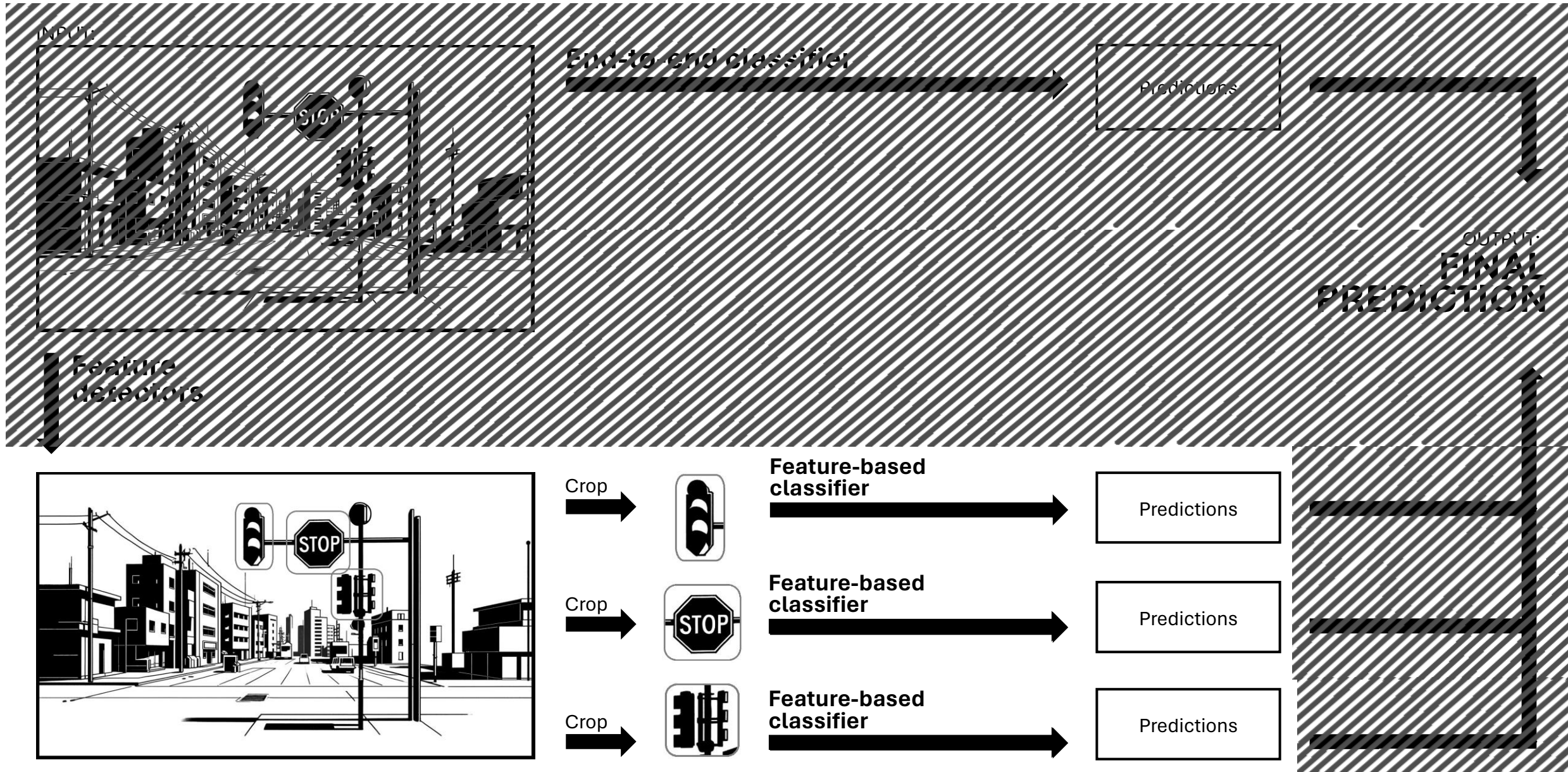


Predictions





# FEATURE-BASED CLASSIFIERS









# FEATURE-BASED CLASSIFIERS

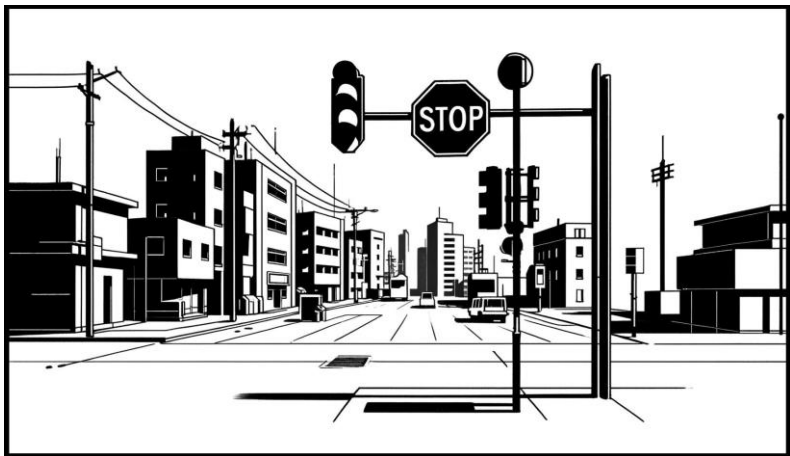
## ■ Basic CNN model:

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 2D)	(None, 25, 25, 16)	0
conv2d_3 (Conv2D)	(None, 25, 25, 32)	4640
max_pooling2d_3 (MaxPooling 2D)	(None, 12, 12, 32)	0
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 the specified feature
- Not enough training data right now to be able to satisfactorily train more complex model architectures

Feature	BASELINES				PERFORMANCES ( ACCURACY )	
	Baseline (top k)	Top 1	Top 2	Top 3	Top 1	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{Traffic Light} \right) \right\}$	0.169	0.333	0.484	0.264	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{Stop Sign} \right) \right\}$	0.263	0.391	0.495	0.363	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{Motorcycle} \right) \right\}$	0.325	0.486	0.601	0.444	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{Car} \right) \right\}$	0.128	0.253	0.360	0.208	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{Bus} \right) \right\}$	0.349	0.425	0.494	0.416	
	$\max_{i_1 < \dots < i_k} \left\{ \sum_{r=1}^k \mathbb{P} \left( \text{CITY}_{i_r} \mid \text{One-way Street} \right) \right\}$	0.142	0.241	0.321	0.208	

INPUT:



End-to-end classifier



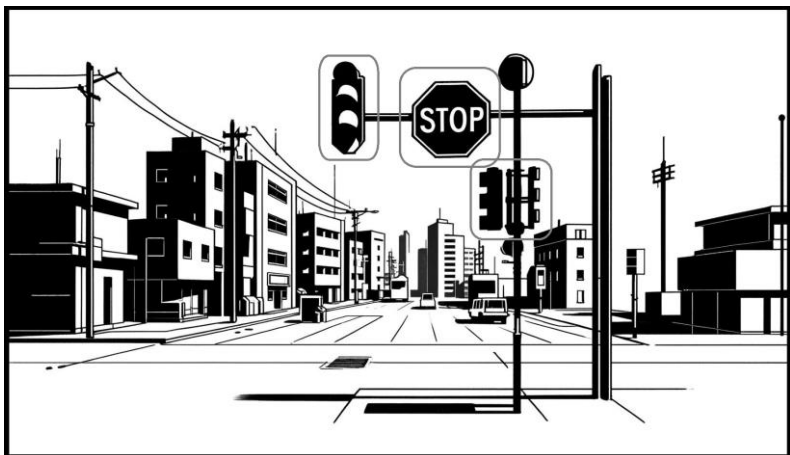
Predictions



OUTPUT:

**FINAL  
PREDICTION**

Feature  
detectors



Crop



Feature-based  
classifier



Predictions

Crop



Feature-based  
classifier



Predictions

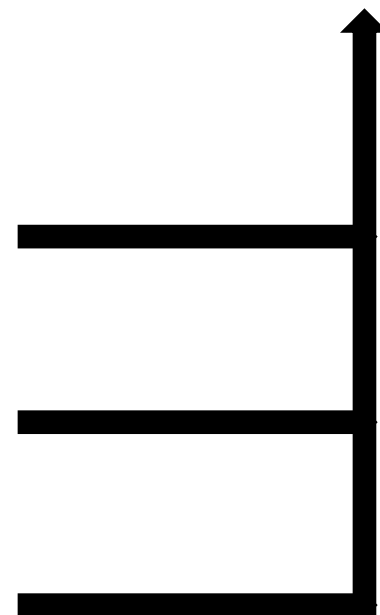
Crop



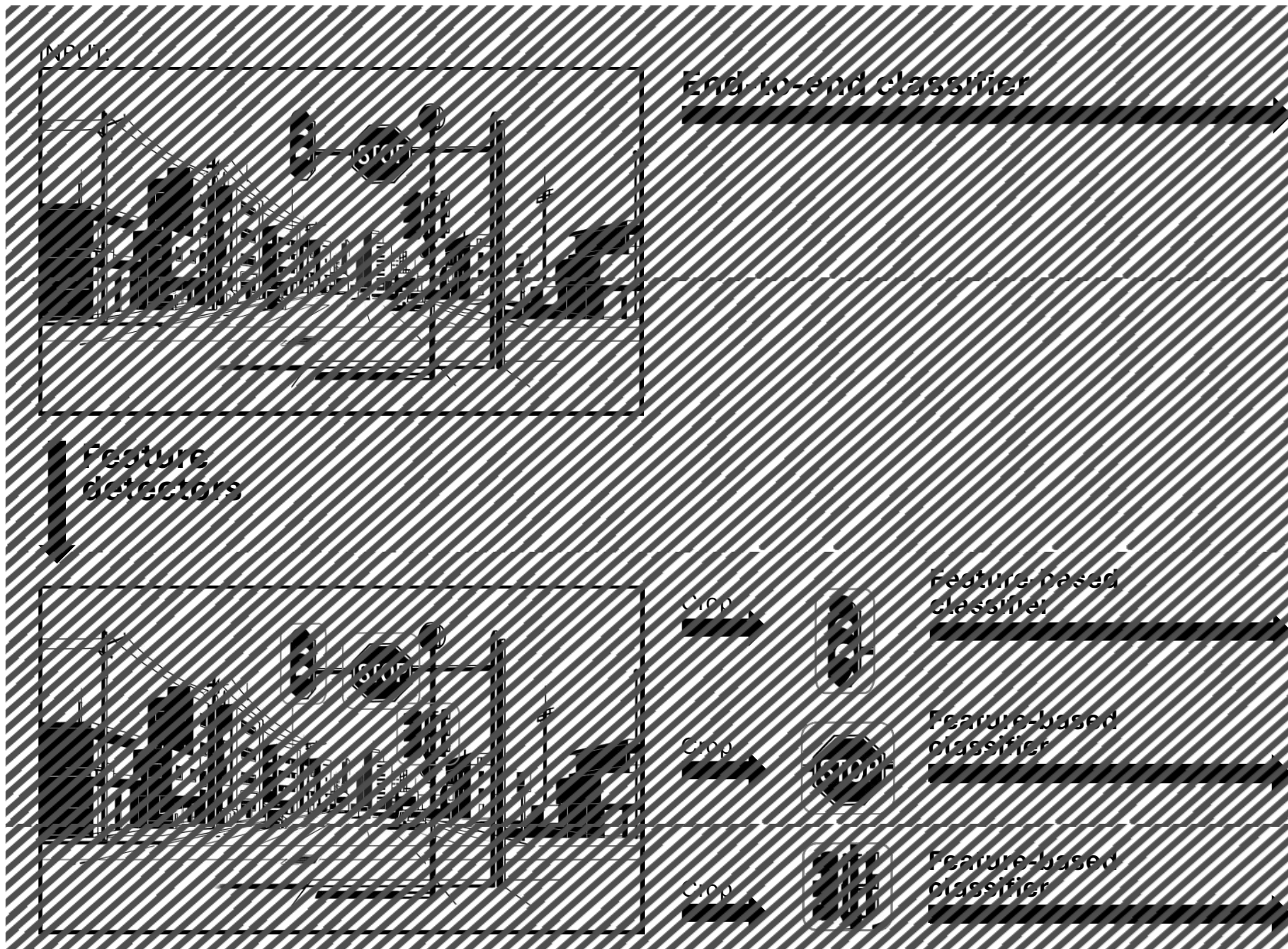
Feature-based  
classifier



Predictions



# ENSEMBLE



Predictions

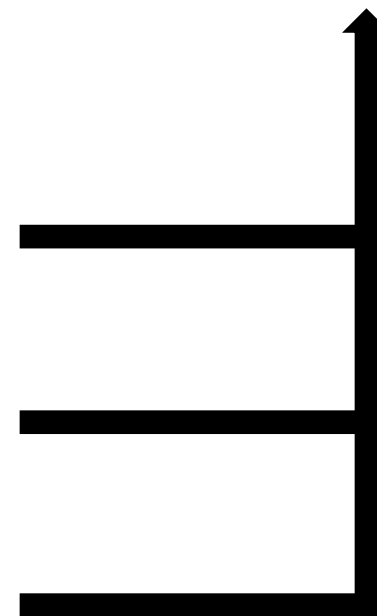


OUTPUT:  
**FINAL  
PREDICTION**

Predictions

Predictions

Predictions



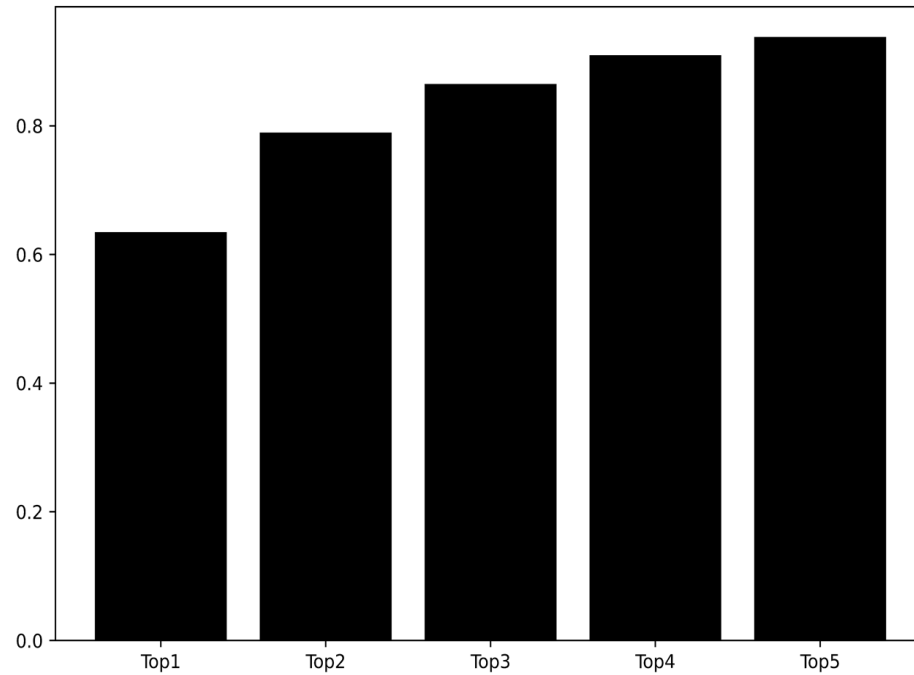


# RESULTS

# PERFORMANCE ANALYSIS

■ Performance results of the complete pipeline according to various metrics

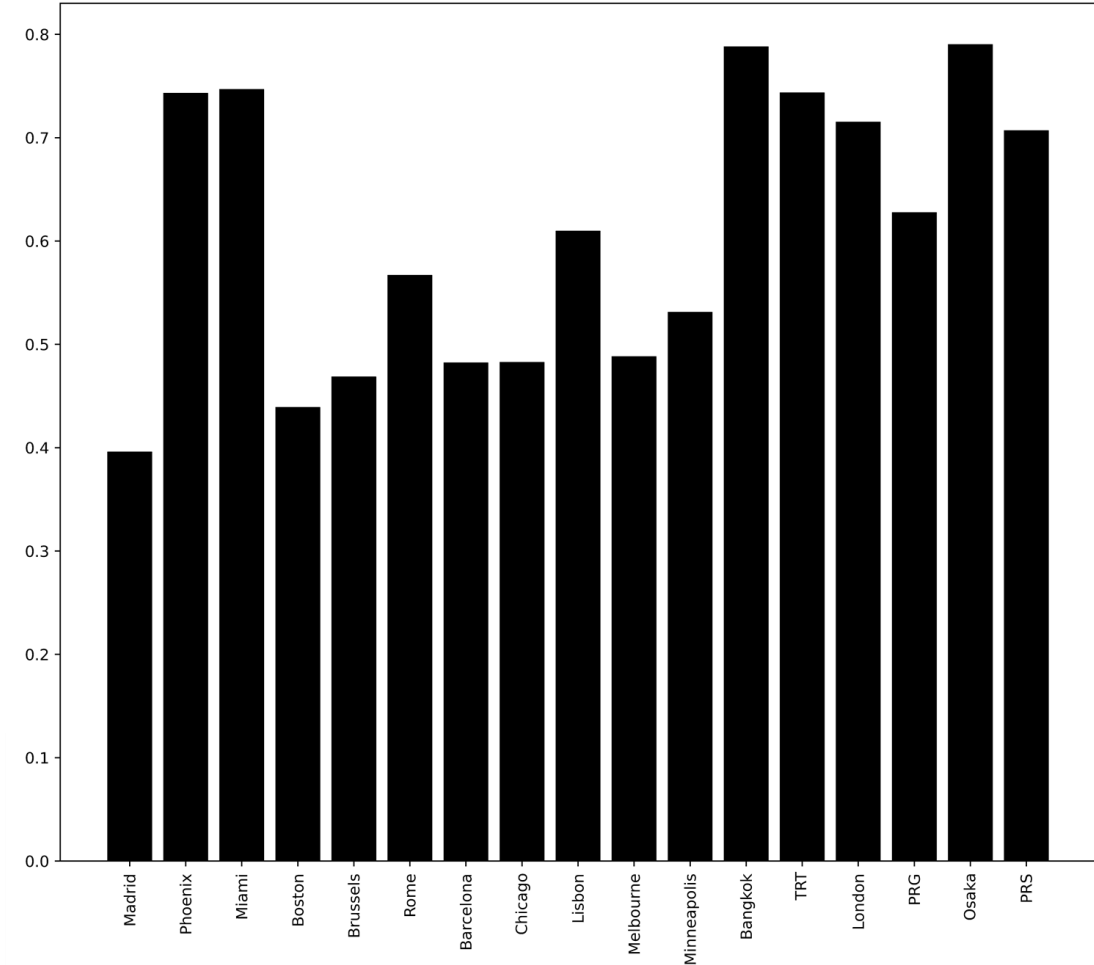
## ACCURACY



**FINAL ACCURACY: 0.635**

( essentially the end-to-end model )

## F1 SCORES





# 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.

MADRID



ROME



# THANK YOU

MADRID



ROME



**DEMO AVAILABLE AT**  
<https://github.com/hochfilzer/geo-locator>