Aviv technical test

Duplicate detection

Hello !

In this presentation we're going to talk about:

- The problem statement
- The AI models proposed
- Results found
- Conclusion and possible improvement

Problem statement

The most simple tactic to bump up listings on top of the flow is to simply spam the marketplace. Every time an ad is posted it will appear on top of the page, making it more likely to get seen by potential buyers or renters. Unfortunately, it will also create a pretty large number of duplicates.

Our goal is to automate the process of **finding duplicates using Artificial Intelligence.**

To do so we are presented with a dataset that contains informations about each ad : **Pictures, text, and other details**

The AI models proposed

Image Matching using SIFT

SIFT is an algorithm used in Computer Vision to extract image main features. We apply the algorithm on both images that we want to compare. And then using Brut Force matcher we score the similarity between them.

Image Matching using Object Detection

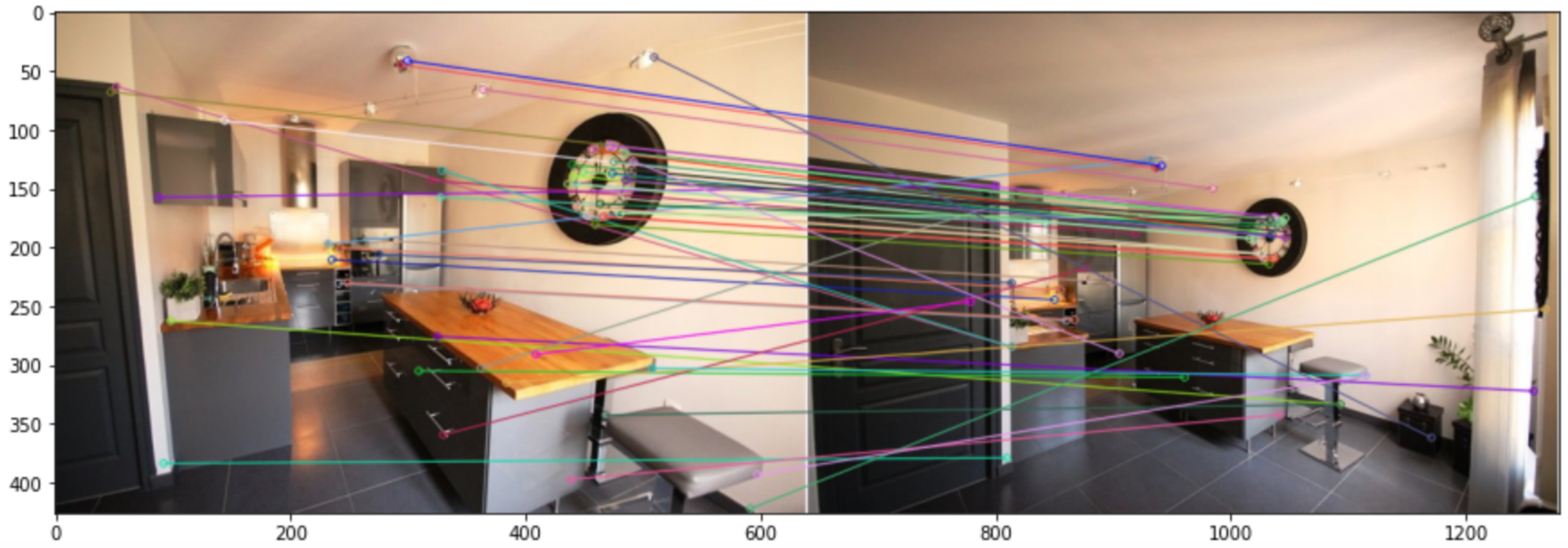
Here we use a pre trained model from the library transformers to detect objects in our images. And then using SIFT algorithm we calculate the similarity between the detected objects in both images and score the similarity

Text Matching using Embedding

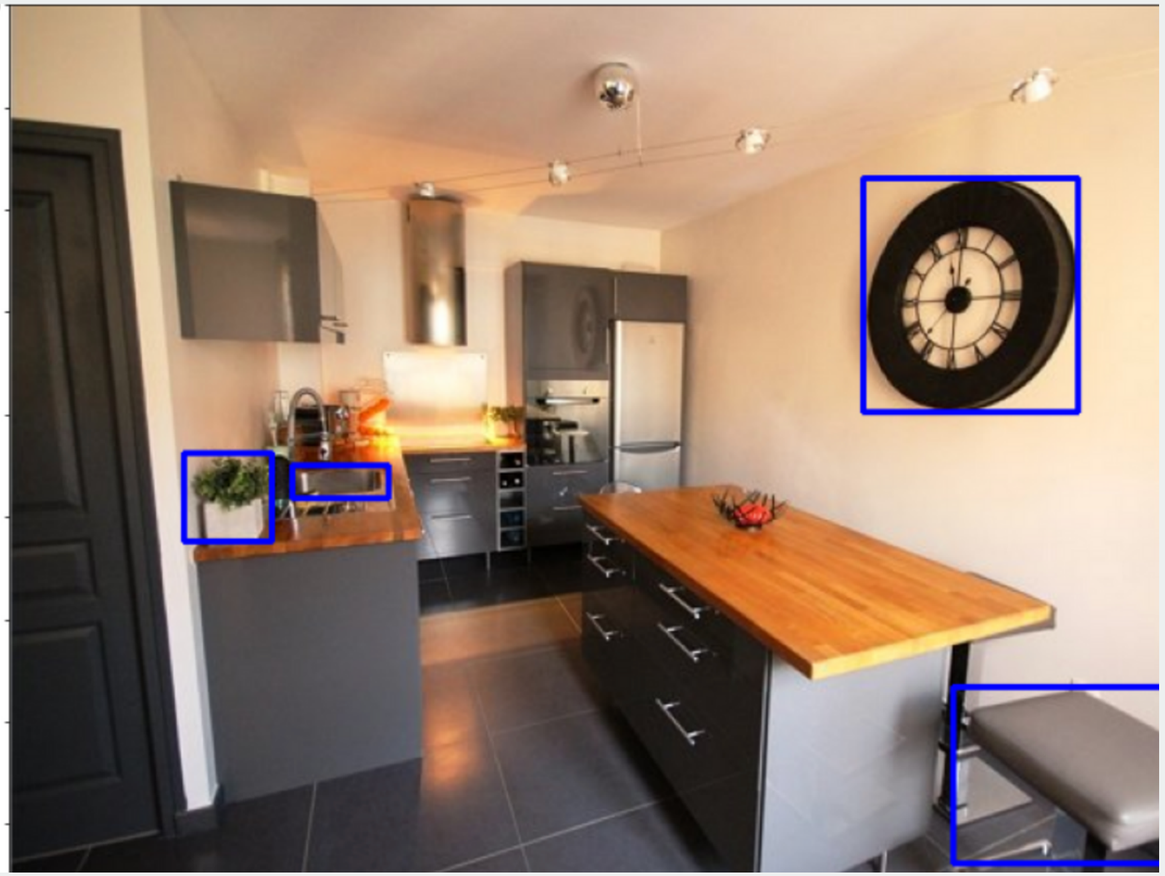
Here we also used a pre trained Embedding from the library transformers to transform our text data into vectors that represent their meaning and then we compare the two inputs using Cosine similarity function

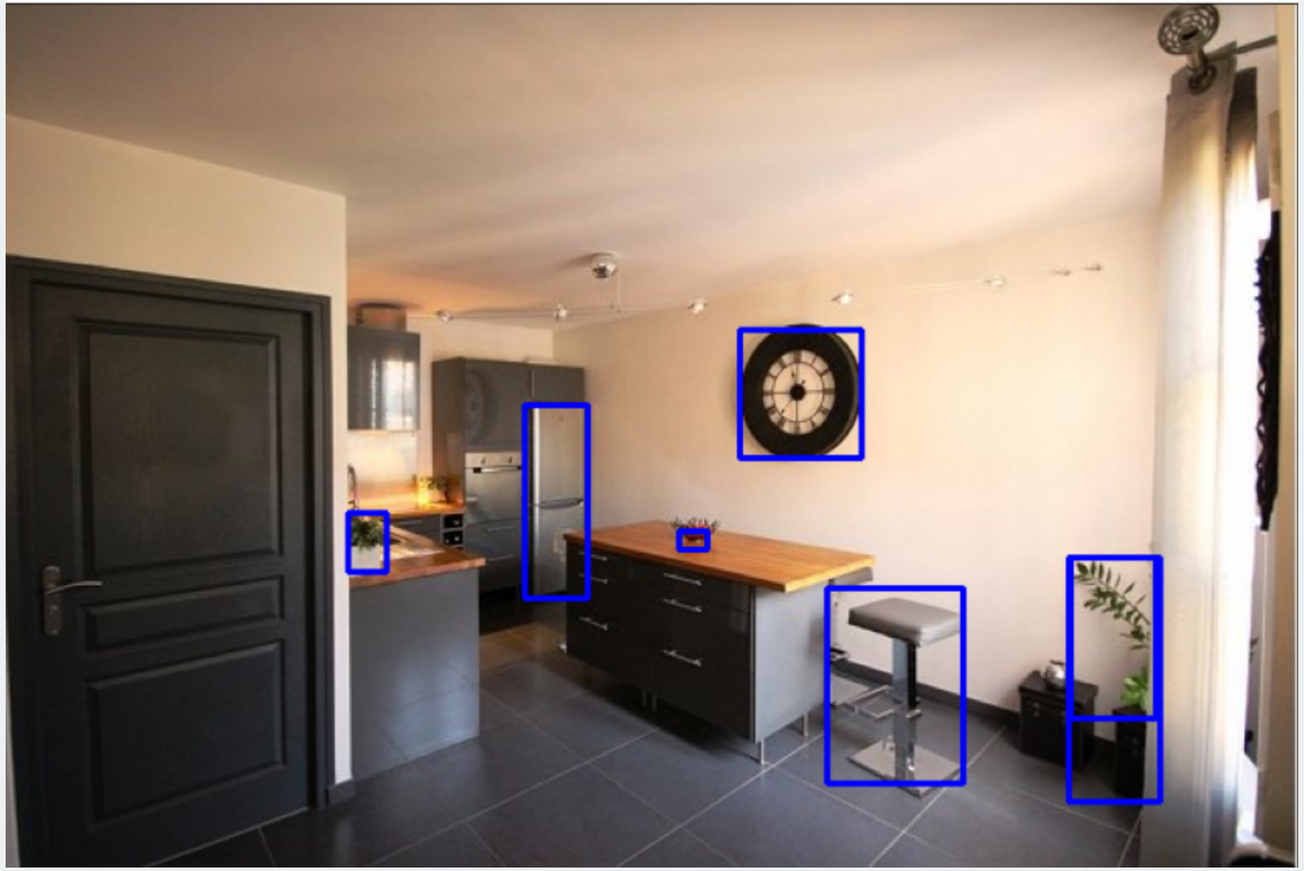
Results

```
Good matches :  56
Total Keypoints :  425
How likely:  13.176470588235295 %
```



Results





Results

	listing_id_1	listing_id_2	labels	pred
0	120777696	116630376	1	[[tensor(1.0000)]]
1	120793420	121050028	1	[[tensor(0.7345)]]
2	118823311	111300261	1	[[tensor(0.7130)]]
3	112597318	102266138	0	[[tensor(0.6553)]]
4	72795989	68729895	1	[[tensor(0.9986)]]
5	116948417	118789539	1	[[tensor(0.9472)]]
6	119253235	35030927	0	[[tensor(0.2783)]]
7	71958413	64731203	0	[[tensor(0.5571)]]
8	115234728	116548598	1	[[tensor(1.)]]
9	122684207	118198070	0	[[tensor(0.4381)]]
10	112597318	106937080	1	[[tensor(0.8995)]]
11	114563510	114768093	1	[[tensor(1.)]]
12	68728690	64730461	1	[[tensor(1.0000)]]
13	122319269	113380488	0	[[tensor(0.6841)]]
14	102299455	99715246	1	[[tensor(0.9675)]]
15	115376555	117949689	1	[[tensor(1.0000)]]
16	121337825	99708641	0	[[tensor(0.6183)]]
17	64723003	105388592	0	[[tensor(0.6260)]]
18	105409752	105417595	0	[[tensor(0.6197)]]
19	116382348	120094263	1	[[tensor(0.6275)]]

Conclusion

- Due to heavy computing I wasn't able to run the algorithms on a sufficient amount of data to evaluate correctly the performance of the models. However from few manual analysis I was able to notice that :

- The text similarity model does a good job in identifying duplicates (even when the paragraph is changed) and it's a good metric for calculating total matching score

- For the SIFT algorithm works with real duplicate images, however when changing the perspective of the image it has more trouble and is harder to scale

- The Object Detection somewhat improves the results when object are correctly detected and visible.

Possible improvements

- I would have spent more time evaluating and tuning the models to improve the results
- I would have used the Siamese Neural Network model, for that I would be needing an annotated dataset with what is considered a duplicate and what not (img to img).
- I would have applied more preprocessing to the images (remove brand names etc)