Cityscape Image Processing and Urban Life Quality Feedback

Aydın Uzun & Gökberk Erdoğan
2015401210 & 2015401099

I. Darwin in the City

As the era of technology blossoms from the roots of the 20th century, the heavy boom of population in major cities became inevitable. However, as people shape their city, the city started to affect their lives and everyday activities, which means that our habits and behaviour evolves as city transforms. The physical appearance of cities, and its perception, impacts the behavior and health of their residents. [1] Within this context, the term behaviour embodies a few major points, namely education, mobility, health and criminal behaviour. Thus, it makes sense to continue shaping our cities according to the alterations and developments regarding the points mentioned.

In order to do that, substantial amount of data needs to be gathered and processed, which is simply neither feasible nor practical through human efforts. Nevertheless, this task becomes more convenient through developing computer vision algorithms trained with human evaluated/labeled data that will conduct evaluations on the images of streets. Hence, in this project, our goal is to implement urban image processing for city shaping.

We believe that this work will enable further progress on city and urban planning and crowd management in addition to global studies of the social and economic effects of architectural and urban planning choices. In addition to that, it can lead to new public transport solutions. Furthermore, eventhough it is a much harder goal even to think of, an application can be designed for smartphones from which the habitants of the city and the tourists could benefit from. The app could simply present daily data, e.g. reporting about the denser parts of the city so that the user could make his/her arrangements accordingly. This would directly regulate the crowd, relieving the pressure on some popular parts of the city and lightening up some serious amount of traffic. However, at this step, this idea is still not feasible at all, since it requires substantial amount of equipment in addition to extremely fast and powerful systems.

A. Dataset Collection & Understanding

The first step of the project is "Dataset Collection & Understanding". It simply refers to gathering images and labeling them for the purpose of training the neural network of the implementation. It is also possible to divide the collected dataset into two parts, one for training and one for testing. Place Pulse is a good example in this context.

Place Pulse is a web interface that collects pairwise image comparisons labeled by humans themselves to collect a dataset for training a convolutional neural network [1] [http://pulse.media.mit.edu]. Place Pulse asks the user/player to compare the two images presented and decide on which one better fits the label given, these labels uses 6 labels, these are safe, lively, boring, wealthy, depressing, and beautiful. Place Pulse 1.0 contains 200,000 pairwise comparisons for 4,109 images from 4 cities. Place Pulse 1.0 has a significant finding. It turns out to be that individual preferences for urban appearance were not driven by participants' age, gender, or location, indicating that there is no significant cultural bias in the dataset. However, a global dataset is still a necessity in order make the program globally accurate,meaning that more instances from more different locations and scenes are required. To that end, Place Pulse 2.0 contains 1.17 million pairwise comparisons for 110,988 images from 56 cities from 28 countries across 6 continents, scored by 81,630 online volunteers. PP 2.0 has significantly higher visual diversity, and hence it improves accuracy by 7.2% as compared to PP 1.0.

At this point, one may ask why satellite images are not preferred over street view images. It is only reasonable to think that satellite images basically covers a much larger area and therefore would be much more efficient and practical. Well, the first difficulty of using satellite images is labelling. It would be a difficult task for a human to consider a place, say, safe, depressing etc. just by throwing a glance at its satellite image. Plus, the cross correlation between the labels are more intense for street view, for example, a "lively" label for a street image could relate to many hints regarding other labels, but that is not valid for satellite images, as it presents the info only on a two dimensional extent.

The article [1] relates to 4 aspects. First one is predicting perceptual responses to images, which considers aesthetics, memorability, interestingness, and virality. The second is the concept of using urban imagery to understand cities. As far as understanding cities is concerned, the article states that especially new sources of data from photo sharing websites and commercial providers enabled further progress on urban planning, architecture and sociology. Relevant to that, the third aspect is understanding the connection between urban appearance and socioeconomic outcomes, which encapsulates health and crime issues. Such an implementation of city scape evaluation may help us confirm whether "Broken Windows Theory" holds or not, i.e. the connection between the perception of unsafety and alcoholism, obesity and STDs can finally be clarified, if there is any. And the last point the article relates to is generating image rankings and comparisons based on semantic and subjective attributes, relative attributes of objects/scenes.

The article [1] mentions two ways for implementing such an algorithm: Streetscore-CNN and Ranking Streetscore-CNN. Streetscore-CNN takes two images, extracts their features using identical tools for both, and then concatenates the features by moving onto the convolutional neural network. Ranking SS-CNN differs from SS-CNN in that it learns an additional set of weights—in comparison to SS-CNN—for minimizing a ranking loss;

$$L_c = \sum_{(i,j,y)} \sum_{k}^{K} [y = k] log(g_k(x_i, x_j)) \tag{1}$$

Both of these methods are two step processes, with the first step being image ranking and the second step being image

classification. Unfortunately, there are some limitations: they require many comparisons and they completely ignore the visual content of the image. Adressing these limitations, the article claims that it would be better to use visual content in the image ranking stage itself by learning to predict pairwise comparisons directly and to train the neural network using image pairs and their crowdsourced comparisons.

The problem formulation in article [1] is as follows(for further reference):

The dataset consists of M images $I = (x_i)_{i=1}^{M}$ and N comparison triplets $P = ((i_k, j_k, y_k))_{k=1}^{N}, y(+1, -1)$. y=+1 indicates for image i, and y=-1 indicates for image j. Hence, the ranking function is:

$$y \times (f_r(x_i) - f_r(x_j)) > 0 \tag{2}$$

B. Feature Detection and Extraction

First of all, one should define what feature means for an image. Roughly speaking, an interesting part of an image can be called as a feature. Defining, understanding and implementing of features is very important, because this is the starting point of many computer vision algorithms. The success of other parts like recognition depends on the success of feature detection.
The main goal of feature extraction is to obtain the most relevant information from the original data and represent that information in a lower dimensionality space. The term 'most relevant' is the critical point. It says that you should get rid of the redundant data. In this context the input data will be transformed into a reduced representation called features vector. This process is called feature extraction. The features vector includes information in the form of isolated points, continuous curves or connected regions.[2] After completing the dataset collection and labeling, feature detection is the first operation performed on an image. Each and every pixel is checked whether or not there is feature present at that pixel thus feature extraction is considered as a low level algorithm.[2]

Figure 1 is a perfect description all the works done in computer vision. In our project we are going to handle only the image processing, feature detection and then recognition parts.

As one can see on the figure 1, a high amount of image preprocessing techniques like binarization, thresholding, resizing, normalization etc. are applied on the sampled image before getting features. After that, feature extraction techniques are applied to get features. After finding out the features the same type of feature have to found out in another image. Therefore the features found out should be described so that they could be matched with other images. This process is called Feature Description. To sum up the processes, first you find out the features and then describe them so that you can match them with other images.

1) Types of image features:

a) Interest points / Keypoint features: The terms corners and interest points are used to some extent interchangeably and refer to point like features in an image. The name "corner" is used in tradition because of early algorithms. First it's performed edge detection and then the edges are analysed to find rapid changes in direction which resulted in corners. In later algorithms there is no need for explicit edge detection.[2]
There are two main approaches to find interest points and their correspondences. The first one is to track features accurately using a local search technique such as correlation or least squares. This approach will be useful if the images are taken from nearby viewpoints or in successive times. The second one is to to detect

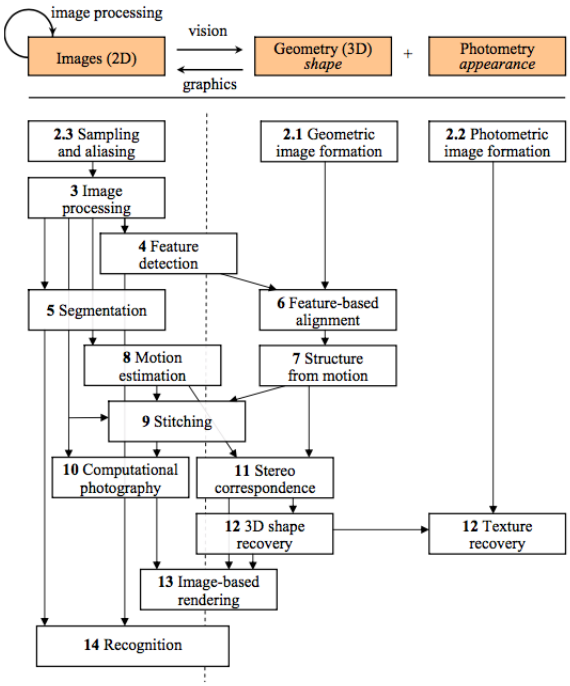

Figure 1.11 Relationship between images, geometry, and photometry, as well as a taxonomy of the topics covered in this book. Topics are roughly positioned along the left–right axis depending on whether they are more closely related to image-based (left), geometry-based (middle) or appearance-based (right) representations, and on the vertical axis by increasing level of abstraction. The whole figure should be taken with a large grain of salt, as there are many additional subtle connections between topics not illustrated here.

Fig. 1: taxonomy of the topics in computer vision [3]

features independently in all images and then match them based on their local appearance. This approach will be useful when a large motion can happen.[3,6]

b) Edges: Qualitatively, edges occur at boundaries between regions of different color, intensity, or texture. A reasonable approach is to define an edge as a location of rapid intensity variation. A mathematical way to define this variation and its direction is through the gradient of image. But it's not so easy at all. There are some problems to take into account. Taking image derivatives amplifies noise. It is necessary to smooth the image with a low pass filter before computing its gradient. Edge detector should also be independent of orientation and therefore a circularly symmetric smoothing filter, gaussian filter, is desirable. This filter is used in most of the edge detection algorithms.[3]

c) Blobs/ regions of interest points: Blobs provide a description of an image structure in terms of regions as opposed to interest points/corners that are more point like. One advantage of blob detection is the fact that one can detect smoother areas which can not be detected by corner detectors. Determinant of Hessian (DoH) and Laplacian of Gaussian (LoG) are two important blob detectors.[3]

d) Ridges: For elongated objects the concept of ridges is a natural tool. From a practical viewpoint, a ridge can be thought of as a one-dimensional curve that represents an axis of symmetry. But extracting ridge features from general classes of grey-level images is very hard algorithmically compared to extracting edge-, corner or blob features. [4]

2) Ideal feature description: What points to choose to extract your feature vector? there are a couple of things to take into account. First, the gradients in at least two significantly different orientations,corners, are the easiest to detect. On the other hand textureless patches are nearly impossible to localize.

Therefore it's meaningful to start with patches with large contrast changes(gradients). [6] But what points to choose and what image features you are going to use depends on completely what you want to achieve. But approaching to ideal case is always rational. An ideal feature descriptor should be compact, effective, distinctive, robust and invariant. [6]

3) Algorithms: There are a lot of feature detection techniques. I tried to emphasize the ones closer to what we discussed in our meeting.

a) SIFT: It stands for Scale Invariant Feature Transform. SIFT is a feature detection algorithm to detect and describe local features in images. Keypoints of objects are first extracted from a set of reference images and stored in a database. Based on Euclidean distance of their feature vectors an object is recognized in a new image by comparing each feature from the new image to the database cerated one step before. After that location, scale and orientation of the object that matches the keypoints from database are determined and filtered out. Each cluster of three or more features agreeing on an object and its pose is then subject to further detailed verification. In the last step the probability that a particular set of features indicates the presence of an object is computed. The matches from the beginning passing all these tests can be identified as correct confidently.[2]

b) HOG: It stands for histogram of oriented gradients. In object detection this method is widely used. In first step the image is divided into small connected regions called cells and its gradient histogram is compiled.[2] After that the results can be contrast-normalized for improved accuracy and better invariance to changes and a decriptor should be returned. The descriptor is the concatenation of these histograms.
The HOG descriptor has a few key advantages over other descriptors. It operates on local cells which means it is invariant to geometric transformations except in terms of object orientation.

4) Bag of Words(BoW) Model: In computer vision bag of words model can be applied to image classification, by treating image features as words.[7] A bag of words is a vector of occurrence counts of local image features. Another definition of the bag of words model will be the "histogram representation based on independent features". First, the words in images need to be defined. To achieve this a three step process should be implemented : feature detection, feature description, codebook generation. In this report it's already told about the feature detection and description steps in detail. Codebook generation is the final step for the bag of words model. In this step the vector represented patches are converted to codewords. This is exactly analogous to words in text documents. A codeword is a representative of several similar patches. After assigning of codewords the image can be represented by the histogram of the codewords. [7]

C. Classification

Classifying data is a common task in machine learning, computer vision and also in our project. We obtained the most relevant information from all dataset and represented them with feature vectors. Now we have to determine whether or not the image contains these specific feature we defined before. There are a huge amount of algorithms used in this field. But the best algoirthms for such tasks are based on convolutional neural networks. We have talked about CNNs, support vector machines and boosting algorithms in our meeting. Therefore I mentioned only these algorithms.

1) Convolutional Neural Networks (CNNs): CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters besides very fast. In contrast, these filters were hand-engineered in traditional algorithms.[10] This is its major advantage. But it's beyond of our lecture EE 475 and therefore we don't use any CNNs while classifying our images. From the beginning of 2012 CNNs are tested and used in image classification in academic world. Especially when applied to facial recognition, CNNs achieves a large decrease in error rate. [10] By the way in today's world the steps feature descriptions and extraction and classification is somewhat combined in one step with the help of CNNs.

2) Support Vector Machine (SVM): Instead of using a neural network to classify patches a support vector machine (SVM) can be used to classify patches.[3] A visual explanation will make things more clear.

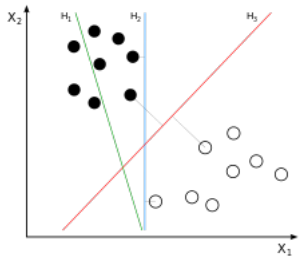

Fig. 2: A model to understand SVM [9]

As one on the figure 2 see, there are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation between two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. This hyperpane is called maximum margin hyperplane and a SVM searches for series of these planes in feature space between different classes. For example between face and nonface patches. One problem occurs when the different classes are not separable in feature space. In this case the feature space should be lifted into higher dimensional features space using kernels.[3]

3) Boosting: Concept of boosting involves training a series of increasingly discriminating simple classifiers and then blending their outputs.[3] In more detail, boosting involves constructing a classifier as a sum of simple weak learners where each of the weak learners is an extremely simple function of the input, and hence is not expected to contribute much to the classification performance. [3] In most variants of boosting, the weak learners are threshold functions, which are also known as decision stumps. Unsurprisingly an explanation with a figure can make things more clear. After each weak classifier is selected,

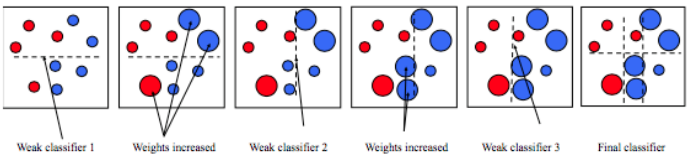

Fig. 3: Schematic illustration of Boosting [3]

data points that are not classified correctly, have their weights increased. The final classifier is a linear combination of the simple weak classifiers.

The method of incrementally selecting the weak learners and reweighting and training examples after each stage as on the figure 3 is the underlying key to success of boosting. The AdaBoost algorithm does this. It reweights each sample as a function of whether it's correctly classified at each stage as on the figure 3 or not. After that it uses the stagewise average classification error to determine the final weighting among week

classifiers. In AdaBoost algorithm the resulting classifier is very fast, but training time is more than in other algorithms. Because there is a lot of things to examine at each stage.

D. Performance Evaluation

The performance of the algorithm depends on a few factors[11]:

(1) the algorithm itself,
(2) the nature of images used to measure the performance of the algorithm,
(3) the algorithm parameters used in the evaluation,
(4) the method used for evaluating the algorithm.

In addition to those, the performance of the implementation could be evaluated on 6 different scales. These include[12]:

(1) accuracy: how well the algorithm has performed with respect to some reference;
(2) robustness: an algorithm's capacity for tolerating various conditions;
(3) sensitivity: how responsive an algorithm is to small changes in features;
(4) adaptability: how the algorithm deals with variability in images;
(5) reliability: the degree to which an algorithm, when repeated using the same stable data, yields the same result;
(6) efficiency: the practical viability of an algorithm (time and space).

E. Presentation

Presentations are one form of communication. Although this is a simple definition, it's one of the building blocks shaping today's world. You may know everything about what you present perfectly, but if you don't present your knowledge well to others if you fail to convince the audience, your knowledge makes no sense. To present your knowledge well is not only a must in academic world, it is a must in all areas of today's world. We are going to make our presentation in the end of our project with this consciousness.

F. Discussion and Future Work

As has been stated at the very beginning, future works could be on improving:
(1) Security,
(2) Mobility,
(3) Social, Economic, Architectural Studies,
(4) Daily Life.

(1): Detecting areas with high crime rate could motivate relevant authorities to take action on determining the characteristics of areas with high crime rate. Further works are then going to consider those characteristics and avoid them.
(2): Again, the authorities will be able to detect which parts of the cities attract the most people and know whether those parts are wealthy, lively and etc. With that info at their disposal, they are going to develop mobility solutions that are fit to their applied areas.
(3): For social, economic and architectural studies, a specific area, location or even streets could be an input to the algorithm so that the argument can label the scenes. Depending on the labeling results, further analysis becomes available at a much large scale.
(4) A smartphone application can be designed at the residents' disposal, however, this would require real time processing and analysis of the images. For example, a user could check the app each day early in the morning in order to plan his/her day, depending on the density of the traffic or density of people at some specific locations.

II. References

[1] A. Dubey, N. Naik, D. Parikh, R. Raskar, C.A. Hidalgo, "Deep Learning the City : Quantifying Urban Perception At A Global Scale", *European Conference on Computer Vision, 2016* [https://arxiv.org/abs/1608.01769], 12 September 2016
[2] H. Angrish, S. Kaur , "A Survey on Feature Description Techniques", *International Journal of Science and Research (IJSR) Volume 4,Issue 5*,[https://www.ijsr.net/archive/v4i5/27041503.pdf], May 2015
[3] R. Szeliski, Computer Vision: Algorithms and Applications. Springer, 2010.
[4] "Feature detection (computer vision)," Wikipedia, 18-Oct-2018. [Online]. Available: https://en.wikipedia.org/wiki/Feature_detection_(computer_vision). [Accessed: 22-Oct-2018].
[5] "Computer vision," Wikipedia, 20-Oct-2018. [Online]. Available: https://en.wikipedia.org/wiki/Computer_vision#Recognition. [Accessed: 22-Oct-2018].
[6] R. Urtasun, "Computer Vision: Image Features," in TTI Chicago.
[7] "Bag-of-words model in computer vision," Wikipedia, 27-Aug-2017. [Online]. Available: https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision. [Accessed: 22-Oct-2018].
[8] A. Pradhan, "SUPPORT VECTOR MACHINE-A Survey," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 8, pp. 82–85, Aug. 2012.
[9] "Support vector machine," Wikipedia, 21-Oct-2018. [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine. [Accessed: 22-Oct-2018].
[10] "Convolutional neural network," Wikipedia, 20-Oct-2018. [Online]. Available: https://en.wikipedia.org/wiki/Convolutional_neural_network. [Accessed: 22-Oct-2018].]
[11] M. D. Heath, S. Sarkar, T. Sanocki, and K. Bowyer, "Robust visual method for assessing the relative performance of edgedetection lgorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 12, pp. 1338–1359, 1997.
[12] M. A. Wirth, "Performance evaluation of image processing algorithms in CADe," Technology in Cancer Research and Treatment, vol. 4, no. 2, pp. 159–172, 2005.