Predicting positions using Independence Graphs

Volker Strobel

June 16, 2016

Abstract

In this report, we analyze the dependence structure of image features and x, y-coordinates using linear regression, graphical Gaussian models, and vine copulas. The dependencies between color and image location are used to build a predictive model for given image features. We compare the predictive power of the used models using the mean absolute error of the predictions on a hold-out test set. The associated uncertainty in the predictions are studied and indications are given, how the proposed model can be used in a real-world scenario.

1 Introduction & Problem Statement

Computer vision tasks—such as object detection, image restoration, or *localization*–often need a high-dimensional reprentation of a feature domain. This task can be fulfilled using probabilistic graphical models and has been used in a variety of applications.

While many problem are studied in the bivariate case only. However, combining copula modelling with machine learning techniques was longtime neglected. Recently, more and more research is focusing on combining machine learning and regression techniques to tackle high-dimensional regression problems Cooke, Joe, and Chang 2015; Elidan 2013.

In this report, we consider the following computer vision localization problem:

Based on a given patch of an image, we would like to predict, where the patch was taken in a larger image—the map image. While existing approaches extract keypoints of the current patch and the map image, succeeded by finding a homography, these approaches are usually computationally complex and do not allow for in-depth analyses of the problem.

This report examines a multivariate problem consisting of five random variables: average red, green, and blue value of an image patch and corresponding x, y position of the patch in a larger map image. For generating the dataset, image patches from a given map image are generated. These images simulate camera images that could be obtained during a flight with a micro aerial vehicle (MAV). The problem is addressed based on two approaches: (i) modifying the environment (i.e., the map image) to make it better suited for the used

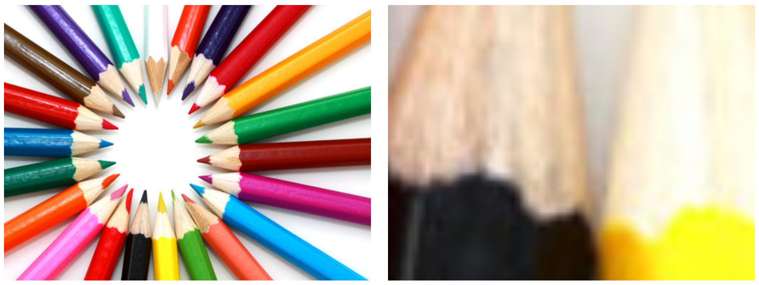

(a) Map image (b) Random image patch

Figure 1: Figure 1a Shows the map image that is used for generating the image patches. Figure 1b shows one sample of the $N = 1000$ image patches that were generated for the generation of the data set.

approach, and (ii) modify the approach to make it better suited to the given map image.

The goal of this report is to select and compare models and use them for predicting the location (x, y-coordinate) of unseen image patches. To this end, we will infer a dependency model that is able to capture the multivariate distribution of the sample data. We compare approaches using linear regression, graphical Gaussian models and vine copulas.

The remainder of this report is structured as follows. Section 2 introduces the method for generating the dataset, and for creating the linear least square predictor, graphical Gaussian models, and vine copulas (we expect the same results for all models – see also corollary Whittaker). Section 3 shows the obtained results. In Section 4, the results are discussed and models are compared. Finally, in Section 5, conclusions and future research directions are given.

2 Methods

2.1 Dataset Generation

Figure 1 shows the process of data set generation. To this end, we will construct a generative model $p(x, y, R, G, B)$ for the distribution. We decided to create the generative model, since it allows to construct an image from the trained model.

In the case of the graphical models, the regression will be performed by deriving a point estimate from the discrminative model $p(x, y \mid R, G, B)$ by using one of *mean, median, mode* of the distribution. Since we will deal with Gaussian distributions, the three point estimates fall together.

By construction, we know that x, y are independent, since the positions were sampled from independent uniform distributions. This leads to the data set that can be found in the appendix.

In this setting, the availability of the data is not a problem and new data can be easily generated. The presented approach is largely data-driven but also exploits the knowledge of the construction of the dataset.

For generating the data set, 500 image positions have been sampled from according to $(x, y) \sim \mathcal{N}(\mu, \Sigma)$, with $\mu = (320, 240)$ (image center) and $\Sigma = \begin{bmatrix} 107 & 0 \\ 0 & 80 \end{bmatrix}$. Therefore, x and y positions are sampled independently. The standard deviations were chosen, such that 99 % of the data points are expected to be in the ranges $[0, 640]$ and $[0, 480]$, respectively.

2.2 First Analysis

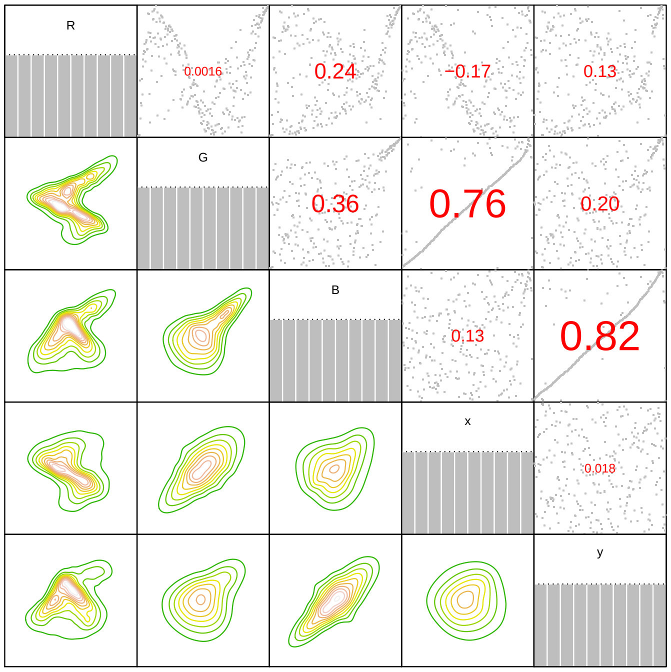


Figure 2: Analysis of pairs of copula data.

In our first model, we use three features: the average red, green, and blue color per image patch. The data set based on $N = 1000$ image patches can be found in the appendix.

The low values of explained variation for x (R^2(x; rest))and y (R^2(x; rest)) show that the used approach is not promising. This is not surprising, given the fact that the colors are non-linearly distributed in the image (see Figure**??**

red	3558				
green	1578	3354			
blue	1219	2749	4016		
x	-1199	-427	-343	22900	
y	-871	-389	616	467	12204
means	180	154	145	244	192
	red	green	blue	x	y

Table 1: The sample variance matrix of the data set

red	1.00				
green	0.46	1.00			
blue	0.32	0.75	1.00		
x	-0.13	-0.05	-0.04	1.00	
y	-0.13	-0.06	0.09	0.03	1.00
	red	green	blue	x	y

Table 2: The sample variance matrix of the data set

for the analysis of the colors). Two alternatives could be used to improve the model's performance. Here, we investigate both.

- Alternative 1: Change the map image such that higher R^2 are obtained
- Alternative 2: Use a more powerful model

2.3 Different map image

In this case, the image was chosen by manual analysis of the features. A more interesting case would be to generate the image based on desired properties of the inverse correlation matrix. An ideal inverse correlation matrix would look like:

One possible solution would be to set all correlations to a high value.

red	1.30				
green	-0.59	2.64			
blue	0.02	-1.81	2.37		
x	0.14	-0.02	0.01	1.02	
y	0.13	0.24	-0.32	-0.01	1.06
R^2	0.23	0.62	0.58	0.02	0.06
	red	green	blue	x	y

Table 3: Inverse correlation matrix

Figure 3: A different map image that makes use of the used features.

red	1.30				
green	-0.59	2.64			
blue	0.02	-1.81	2.37		
x	0.14	-0.02	0.01	1.02	
y	0.13	0.24	-0.32	-0.01	1.06
R^2	0.23	0.62	0.58	0.02	0.06
	red	green	blue	x	y

Table 4: Inverse correlation matrix

2.4 The backward approach

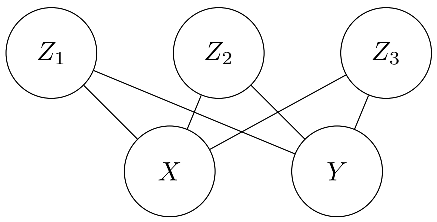


Figure 4: The graphical model used for generating ideal images.

Knowing the criteria for computing R^2, we could construct optimal images for the given approach. These images will be gradient images. Since images consist of three channels, we could encode the information in two channels, and use arbitrary information in the third channel—and therefore even keep our initial image to a great extent.

2.5 Gaussian Graphical Model

In this subsection, we construct a Gaussian graphical model from our sample. Two major steps are involved in this procedure: construction of the independence graph (model selection) and likelihood inference. To this end, we test for independence using the chi-square test and fit the variance matrix using iterative proportional fitting. Since we generated the data, we know that the samples are identically and independally distributed (i.i.d). The following conditional independence statements can be read from the graph.

2.5.1 Model Selection

While a thresholded exclusions strategy (e.g. corr < 0.1) seems directly accessible, it does not capture the size and therefore available information of the sample data.

Therefore, for the model selection, we start with a saturated graphical model and employ the backward elimination with deviance difference stopping rule.

This leads to the model in Figure **??**. This leads to a deviance of 0.13 on 4 df with p-value of x, resulting in the following $\hat{V}$:

2.5.2 Inference

In a first step, the covariance matrix and its inverse are computed. Then the partial correlations are obtained.

2.5.3 Fitting

Now, we use the IPF algorithm to update the sample variance matrix and fit it to the given graph.

2.5.4 Predictions

From Proposition 6.3.1, we know that the conditional distribution of X_b given $X_a = a$ is Normal with mean

$$E_{b|a}(X_b) = \mu_b + V_{ba}V_{aa}^{-1}(x_a - \mu_a) \tag{1}$$

and variance

$$\mathrm{var}_{b|a}(X_b) = V_{bb|a} = V_{bb} - V_{ba}V_{aa}^{-1}V_{ab} \tag{2}$$

In our case, R,G,B are given, while x, y are desired. Therefore, we set $X_a = (R, G, B)$ and $X_b = (x, y)$.

$$E_{b|a}(X_b) = \mu_b + V_{ba}V_{aa}^{-1}(x_a - \mu_a) \tag{3}$$

and variance

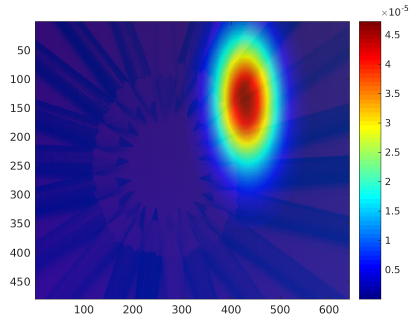
$$\mathrm{var}_{b|a}(X_b) = V_{bb|a} = V_{bb} - V_{ba}V_{aa}^{-1}V_{ab} \tag{4}$$

.

$\mu_b = (320, 240)$ is the center point of the image.

As expected, the conditional mean and the linear least square predictor are identical (see Corollary 6.3.2). The higher variance in the x coordinate of the conditional distribution is directly related to the R^2 value.

The GGM is able to infer the independence between the random variables R and (x, y), leading to more accurate predictions.


(a) Image overlay

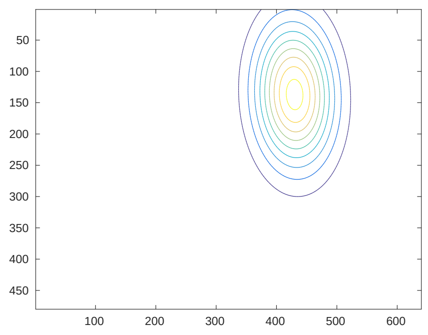

(b) Contour plot

Figure 5: Example of the visualization of the predictions using RGB = (100, 70, 200)

2.6 Vine Copula

In our final approach, we try to capture the dependence structure without modifications of the original map image. In order to do so, we need a more powerful model. The use of *copulas* allows to model marginal distributions and dependence structure independently, allowing for convenient and powerful representation of joint probability distributions. Vine copulas leverage these advantages and bring the advantages to higher dimensional distributions.

The vine copula approach is based on Sklar's theorem that states that multivariate distribution can be described by marginal distributions and the dependence structure—-the copulas. Using bivariate copulas as building blocks, more complex interaction structures can be achieved by building *vine copulas*–nested sets of connected trees.

While it is theoretically possible to get $F(x)$ by evaluating it at $\dots$, in practice we have to calculate the inversion of the (empiricial / pseudo) CDF.

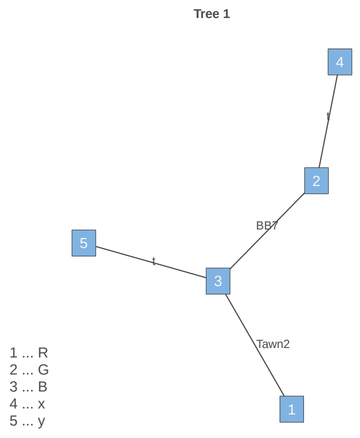
3 Results

A visualization of the used bivariate copulas can be found in the appendix (B).

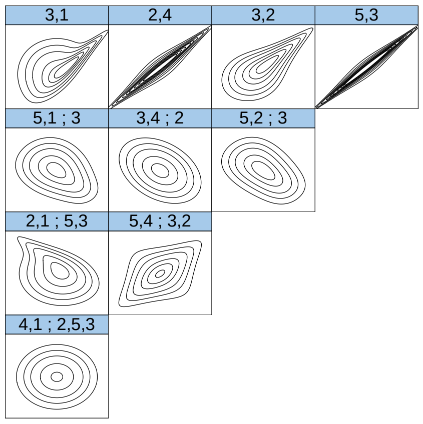
4 Discussion

The presented results indicate the suitability of our approach. In a real-world setting, run-time and computational complexity will be crucial which we left out here.

An interesting future research direction will be the inclusion of time dependence: in a real-world scenario, images will not be i.i.d but highly correlated.



(a) Image overlay



(b) Contour plot

Figure 6: Example of the visualization of the predictions using RGB = (100, 70, 200)

5 Conclusion

In this report, we compared linear regression, graphical Gaussian models and vine copulas for the dependence modeling of variables in a computer vision task.

References

Cooke, Roger M, Harry Joe, and Bo Chang (2015). “Vine Regression”. In: *Resources for the Future Discussion Paper*, pp. 15–52.

Elidan, Gal (2013). “Copulas in machine learning”. In: *Copulae in mathematical and quantitative finance*. Springer, pp. 39–60.

A Supplementary Material

B Visualization of the used copulas

B.1 Tree 1

B.2 Tree 2

B.3 Tree 3

C Dataset with average red, green, and blue values

red	green	blue	x	y
199.15	184.01	148.09	173.21	312.11
161.74	104.92	98.682	471.52	156.44
152.76	115.75	91.875	61.441	70.699
219.71	120.5	92.252	304.45	61.207
221.39	199.51	169.75	164.92	40.368
228.69	224.08	196.25	185.25	259.92
181.83	131.69	147.57	456.16	218.93
195.77	71.636	60.897	32.196	107.07
109.1	193.62	143.95	432.03	95.567
255	255	255	168.08	182.98
234.87	192.93	147.54	473.74	37.964
255	255	255	194.94	135.98
246.92	165.26	213.07	-9.5128	346.47
...	...	...	...	...

D Technical Report

The code for generating the graphs and yielding the results presented in this report was written in `R`. The used data is saved in CSV format. Both can be found on GitHub:
`https://github.com/Pold87/decision-theory`

The section on Graphical Gaussian Models made use of the packages `gRim`, `gRbase` and `Rgraphviz` for the visualization.

The section on VineCopula used the packages `copula` and `VineCopula`.