Machine Learning Techniques for Forensic Footwear Analysis

Thesis

Submitted in Partial Fulfillment of the requirements of: BITS F421T Thesis
By

Gautham Venkatasubramanian
ID No. 2014B4A70637P

Under the Supervision of:
Dr. Martin Herman

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
PILANI CAMPUS
December 2018

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Martin Herman, for allowing me to pursue this project under his supervision, and for his guidance and direction in all facets of my work in this project.

I am grateful to Dr. Hari Iyer and Dr. Steve Lund, who provided many valuable suggestions that helped me gain new insights whenever I was stuck with some problem. Their feedback on my work helped me understand certain nuances of the concepts I was dealing with.

I would like to thank the other members of the NIST Forensic Footwear Team for their reflections on my work during the weekly meetings, which provided me with many ideas to refine my work. I am also grateful to the past members of the team: their work was instrumental in shaping my own.

I would like to thank Dr. Jennifer Ranjani for consenting to be my co-supervisor for this project, and the Department of Computer Science and Information Systems, BITS Pilani, for their assistance and support.

CERTIFICATE

This is to certify that the thesis entitled *"Machine Learning Techniques for Forensic Footwear Analysis"* and submitted by Gautham Venkatasubramanian ID No. 2014B4A70637P in partial fulfillment of the requirements of BITS F421T thesis embodies the work done by him under my supervision.

_____

Dr. Martin Herman
Senior Advisor for Forensics and IT
National Institute of Standards and Technology
Gaithersburg, Maryland USA

Date :

ABSTRACT

We will review current approaches to analysis of footwear impression evidence in the context of one-to-one comparison and unique identification. We will then consider the constraints that would need to placed on the process of similarity in order to have a method that can produce scores with distinguishing power.

Following this, we will study conventional methods of similarity computation on images, starting with the theory of moment invariants up until parametric models. Finally, we will look at some pre-trained deep neural network models and consider their usefulness in feature extraction and comparison.

Contents

Introduction

Usage of Shoeprints in Forensics

Footwear impressions are readily found in crime scenes, and developments in forensic evidence have allowed for the recording of such impressions as images. These impressions are used by investigators in apprehending suspects, and also as evidence in a court of law. As there are a wide variety of footwear models, it is necessary to have reliable methods of finding the type (brand, model etc.) of shoe which can produce such an impression.

Once the type of shoe has been identified and suspect(s) have been discovered, it is necessary to analyze further details to uniquely identify the exact shoe which could have produced a given impression. This unique identification is usually done manually by an expert footwear examiner. The aim of the NIST Forensic Footwear team is to discover how automated methods can aid an investigator in the identification process.

Finding the model of shoe

Finding the model of shoe for a given impression can be described as a database retrieval problem: Given a large database of shoeprint impressions, one has to find efficient ways of retrieving a matching impression from the database for a given query impression(as in Figure 1). We summarize some of the developments in this field.

1995-2002: Early Forays

- The need for database retrieval of shoeprint images is identified; databases like REBEZO (Geradts & Keijzer, 1996) and SHOE-FIT (Sawyer & Monckton, 1995) are developed for a systematic method of recording and retrieving shoeprint information.
- Various features such as points, bars, squiggles, and shapes are manually identified and entered into the database in the form of codes. These codes are used to classify the images, and retrieval is done via regular database queries.

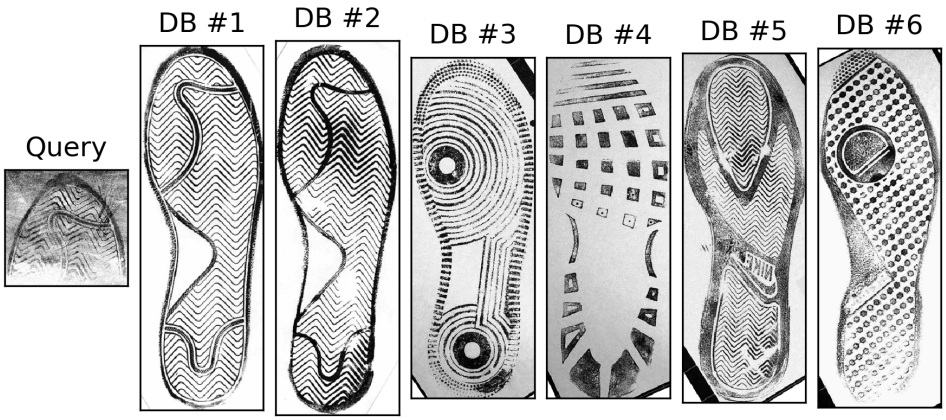

Figure 1: Database Retrieval

- Attempts are made to automate the retrieval: various methods (Fourier transforms, Fractal decompositions (Bouridane, Alexander, Nibouche, & Crookes, 2000) etc.) are tried, but computational aspects hamper the feasibility.

2003-2010: Signal Processing methods

- As various methods for comparing images are discovered, some of these are attempted on shoeprints. A simple description of the retrieval procedure would be as follows:
  1. Transform the queried impression into a feature space via a suitable set of functions (i.e. a filterbank)
  2. Compare it to database images in the feature space and assign a similarity score
  3. Retrieve from the database the image that has the best score when compared to the query impression.
- For assigning the feature space, many different functions were used: Radon transforms (Patil, Deshmukh, & Kulkarni, 2012), Gabor wavelets (Patil & Kulkarni, 2009), Fourier-Mellin (Gueham, Bouridane, Crookes, & Nibouche, 2008) transforms et cetera. For the comparison, the transformed images were either compared directly (i.e. using normalized cross correlation) or via their image moments.

2010-present: Machine Learning and Computer Vision

- Advances in machine learning enabled more flexible methods of feature extraction from shoeprints. This in turn resulted in retrieval methods of high accuracy for a given database, as one could now train an algorithm to accurately retrieve shoeprint images.
- Object recognition models and other discoveries in deep learning are being attempted on shoeprints: most recently, parts of ResNet-50 (Kong, Ramanan, & Fowlkes, 2017) were used to extract features and compare images.

Obtaining one-to-one similarity scores

While the above discoveries are extremely accurate at retrieving a given shoe from a database, it is not known if the exact method(s) used to identify a model from a database can be used to distinguish between two shoes of the same model.

Consider the example in Figure 2: We can see that the third image is not as good a match to the crime scene as compared to the second, but it is not known how such a distinction is expressed quantitatively during database retrieval. My goal is to find out how the performance of database retrieval algorithms varies in such cases.

Variability in Crime Scene Images

Additionally, images collected at the scene of a crime can have distortions that may make comparison difficult. These distortions are including but not restricted to:

- **Translation** - the impression in the crime scene may not be centered, as it is in database images.
- **Rotation** - the impression in the crime scene may not be aligned vertically as a database image.
- **Scale** - Unless recorded along with the image, the shoe size cannot be reliably determined from just the pixel data, thus even if a database image is retrieved, matching the impression to an exact size is difficult.

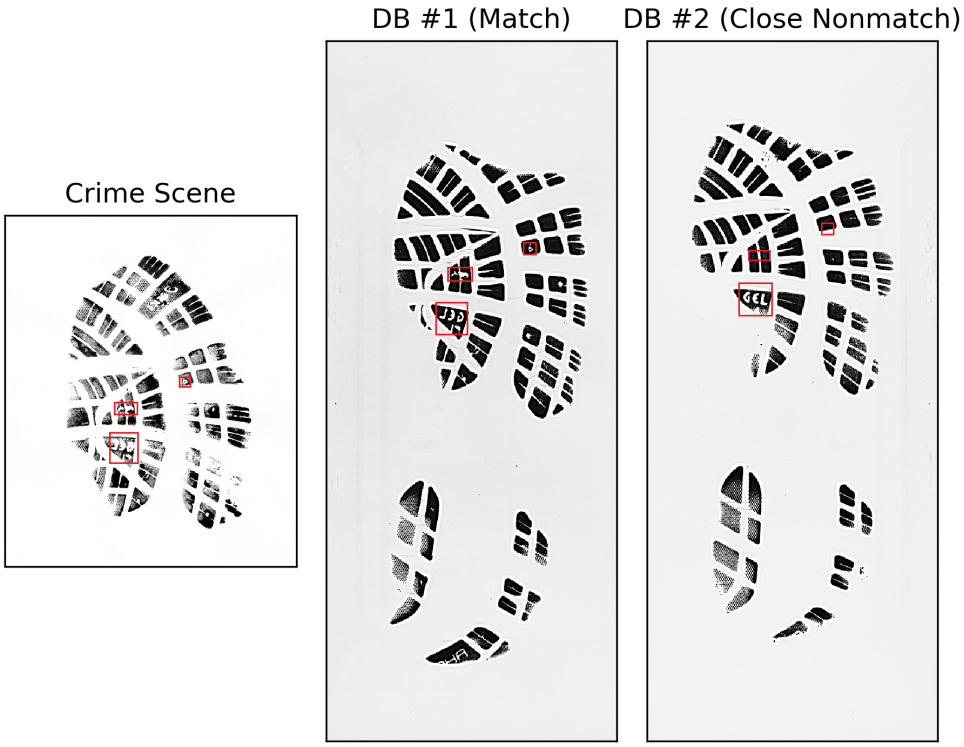

Figure 2: Notice differences in the rectangular boxes

- **Occlusion** - Due to environmental factors, it may be possible to only capture a partial impression.
- **Non-rigid distortions** - Similar to occlusion, the impression may have some non-rigid distortions (stretch, shear) due to how it was made: for example, impressions captured from running are different compared to those captured from walking.
- **Noise** - The quality of the crime scene image is lower than a database impression, because there might be other information (design of the surface, overlapping impressions, measurement marks) that interferes with the shoe information.

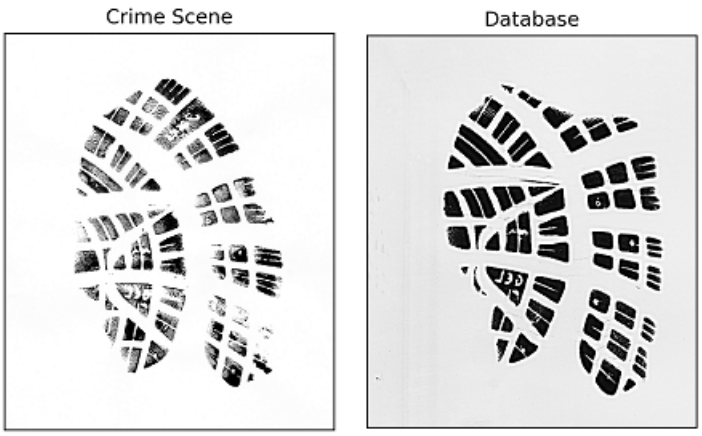

Figure 3: Example of minor distortions

While database retrieval algorithms do provide robustness against some of the above factors, the extent of their stability is not known. Moreover, there is little knowledge of the performance of shoeprint algorithms for obtaining one-to-one similarity scores of crime scene images with given test impressions.

To account for these factors, it was decided that the crime-scene images would have to be manually marked up and aligned, so that the database retrieval algorithms can be tested in a proper manner.

Proposed process

To obtain one-to-one similarity scores between images: the following process has been considered:

1. Manually mark the important parts the crime scene image
2. Use this information to align a crime scene image with a given test impression
3. Compute the features and similarity scores using the various database retrieval algorithm
4. Use a combination of these features/scores to obtain a metric to match two given shoe impressions.

Manual Markup of Crime Scene Images

Types of features marked

As the quality of a crime scene image can vary wildly, it is necessary to have a range of features that can be marked depending on how easy it is to identify them in an image. Once marked, these features can be stored in text or binary format along with the crime scene image, and provide helpful information to algorithms that analyze the image for feature extraction.

The following types of features can be marked on a crime-scene image:

- **Regions of contact**: Even if the image is of bad quality, it may be possible to identify at least the points/regions of the image where one is sure that the shoe made contact with the floor to produce the impression.
- **Regions of non-contact**: Similar to the previous, it may be possible to mark regions of the image where one is sure that the shoe *did not* make contact with the surface. This information provides additional distinguishing power over just marking contact regions. It can also be used to obtain information about the surface on which the impression was obtained.
- **Regions of exclusion**: These regions of the image indicate that there is no shoe-related information present, and can be used to ignore those parts of the image that might interfere with the comparisons.
- **Edges**: If visible, edges (either as line segments or curves) provide strong information about the patterns in the shoe impression. The marked edges may then also be used in obtain similarity scores.
- **Corners**: Corner points provide stronger information than edges about patterns in a shoe, and as such are harder to find in an image of low quality. However, finding many corners is beneficial to alignment of the shoe as well as similarity computations.
- **Polylines, Shapes, and Closed curves**: These complex markings provide the most information about the patterns in a shoeprint image; however, it may not be possible to mark such features completely if the image is of low quality.

A graphical user interface was developed in Python to perform the markups

of shoeprint images as described previously. The marked features are stored in a JSON file.

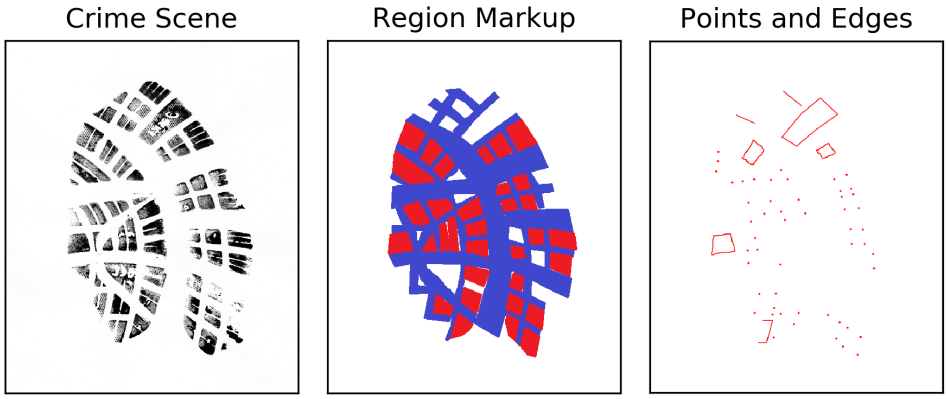

Figure 4: Example markup(red and blue indicates contact and non-contact regions respectively)

Image Alignment

Once the crime scene image has been marked up, the recorded features can be used to align it with a given impression from the database. By alignment, we mean that we find a transformation (rigid or non-rigid as per requirement) to map the crime scene image to the test impression.

As the database impression is of good quality, we must be able to automatically extract from it features that were marked up manually in the crime scene image.

- Regions of contact and non-contact can be found via an image thresholding operation in most cases.
- Edges can be extracted using Canny edge detector (Canny, 1986), along with Hough transforms (Duda & Hart, 1972) if line segments, circles, or ellipses are available.
- Corners can be extracted via methods like Harris Corner Detector (*Harris Corner Detector*, n.d.).

Image Alignment as a Point Pattern problem

If we consider a shoeprint image as a large set of points in contact with respect to a surface, the problem of aligning two images reduces to finding the required transformation to map one of the point sets to (a subset of) the other. However, it is not computationally feasible to consider each pixel of the shoeprint as a contact point. So we either have to consider a sample of contact points or choose points with greater distinguishing power.

Let us consider sampling contact points:

- Let S_1 be a marked-up crime scene image, and S_2 be a database impression. We perform thresholding on S_2 to obtain its contact and non-contact regions.
- We use marked points in S_1 and sample points from the contact regions S_2 for finding an alignment.
- For greater accuracy, we obtain alignments from multiple samples, and consider an average.

The question arises as to how one can pick a *representative* sample of contact points: instead of just random sampling, we consider a sample of points from the image such that each pair of points are separated by a distance of Δ or greater. This ensures that our sampling is well spread across all contact regions of the shoe.

Alternatively, we can choose points with greater distinguishing power, like points on edges or corners. While these points are a (small) subset of all contact points, they provide a more structured representation of the shoe than just sampled contact points, and hence are more suitable for finding an alignment.

To obtain edge points, we can run an edge extraction algorithm on the image and sample points from the response image of the edge detector, because the number of edges pixels may also be substantial. For corner points, we can use a corner detection algorithm.

Accurate detection of corner points

The issue of accuracy arises with the extraction of corner points from an image:

- How do we know that the detection algorithm extracts a high percentage of corners that are actually in the image, i.e. does the algorithm have a high *recall*?
- What is the percentage of erroneous corners extracted by the detection algorithm from the image, i.e. does the algorithm have a high *precision*?

We found that the readily available corner detection algorithms like Harris (*Harris Corner Detector*, n.d.), Shi-Tomasi (*Shi-Tomasi Corner Detector*, n.d.), and FAST (Rosten & Drummond, 2006) do not provide the required amount of precision and recall while finding corners in the shoeprint images. Hence a modification was made to the FAST algorithm to account for this. To assign the corner response for a given pixel, the conventional FAST algorithm considers 16 border pixels of a Bresenham circle of radius 3 (in red) around the pixel. We consider 8 additional pixels (in orange) in the neighbourhood (Figure 5):

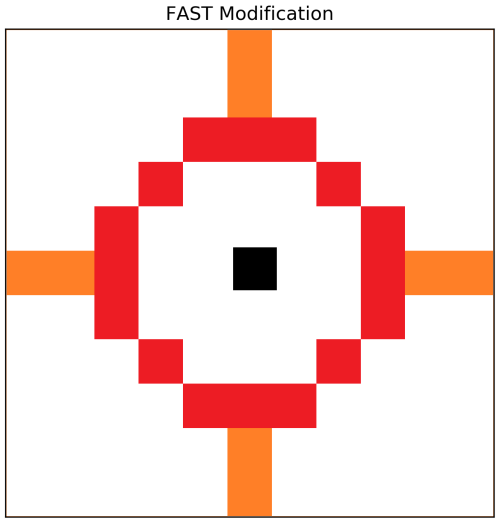

Figure 5: Additional pixels help finding better corners

We also run an averaging filter in order to reduce spurious maxima in the

corner response image. We find that these modifications help increase the precision and recall of corner detection for our use case (Figure 6).

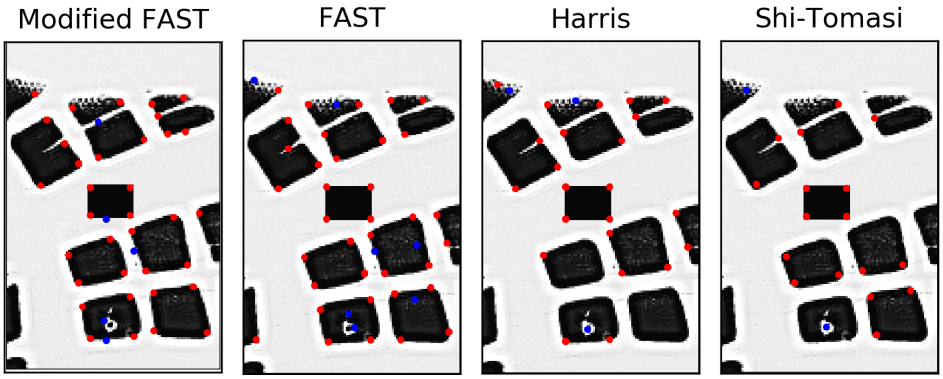

Figure 6: The modification to FAST suits our use case

For further improvements, we can also train a decision tree or neural network to refine the set of corners extracted via the above algorithm, as was done in the original implementation of FAST.

Clustering-based Alignment

Once points of interest have been obtained from the shoeprint images, we can consider the problem of aligning the two given point patterns.

Let S_1 be a shoeprint with M points of interest, and let S_2 be a shoeprint with N points of interest:

$$S_1 = \{p_1, p_2, p_3 .... p_M\} p_i = (x_i, y_i)$$
$$S_2 = \{q_1, q_2 .... q_N\} q_j = (x_j, y_j)$$

We use S_1 to denote both the shoeprint and the set of its interest points. Now, we have to find a transformation T that maps maximum number of points from S_1 to S_2.

First, let us restrict the transformation T to involve only rotation and translation. We have to find an angle of rotation θ a translation vector $\alpha\hat{i} + \beta\hat{j}$ such that:

$$|u_k * T - v_k| \; 0$$

for a maximal number of points $\{u_k\} \subset S_1$, $\{v_k\} \subset S_2$. We have to find the optimum value θ^* and $\alpha^*\hat{i} + \beta^*\hat{j}$ for the above problem.

Let p, q be points on S_1 and S_2 respectively. If S_1 is rotated by an angle θ, we define $t_{p,q}(\theta)$ as the translation vector required to move $p(\theta)$ to q.

Now, we define $T_u(\theta)$ as the set of all possible translations that we can apply on $p \in S_1$ after rotation.

$$T_u(\theta) = \{t_{p,q}(\theta); q \in S_2\}$$

Note that $|T_u(\theta)| = N$ as a point $p \in S_1$ can be mapped to any point of the N points $q \in S_2$.

We define $T(\theta)$ as the set of translations over all points $p \in S_1$ after rotation, i.e.

$$T(\theta) = \bigcup_{p \in S_1} T_u(\theta) = \{t_{p,q}(\theta); \forall p \in S_1, \forall q \in S_2\}$$

Again, note that $|T(\theta)| = MN$.

If we denote the point p as (p_x, p_y) and q as (q_x, q_y), on rotationof p by θ we get

$$p(\theta) = (p_x cos\theta - p_y sin\theta, p_y cos\theta + p_x sin\theta)$$

Since $t_{p,q}(\theta)$ is a 2D translation vector for given values of p, q, θ, we obtain an expression for it as follows:

$$t_{p,q}(\theta) = (q_x - p_x cos\theta + p_y sin\theta, q_y - p_y cos\theta - p_x sin\theta)$$

With $t_{p,q}(\theta)$ as above, for a given value of θ we can plot all $t_{p,q}(\theta)$ as *points in the space of 2D translation vectors.*

In Figure 7, each point in the third plot corresponds to a translation vector $t_{p,q}(\theta)$ between a point $p \in S_1$ and a point $q \in S_2$ (denoted by a line). We calculate the distance between two points using the Euclidean distance metric.

Now, we notice that certain translation vectors (in red) are *clustered together* at a particular location in the above plot. The center of the densest such

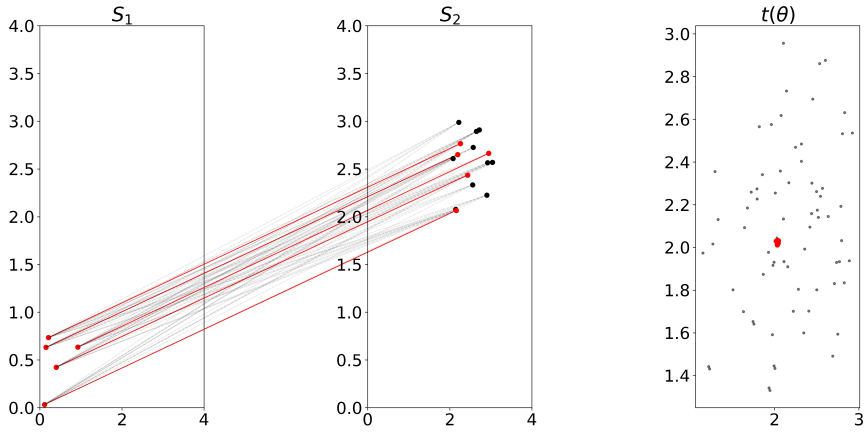

Figure 7: Clustering in translation space

cluster is defined as the optimum translation vector $\alpha\hat{i}+\beta\hat{j}$ for a given angle of rotation θ. The density of the cluster indicates the *strength of correspondence*, i.e. how many points $p \in S_1$ can be mapped with this translation vector to points $q \in S_2$.

Hence for two sets of points S_1, S_2 and rotation angle θ we obtain the following triplet:

$$(\theta, \alpha_\theta\hat{i} + \beta_\theta\hat{j}, d_\theta)$$

We need to find θ such that the correspondence (i.e. d_θ) is maximized. The triplet corresponding to the optimum thus obtained is

$$(\theta^*, \alpha^*\hat{i} + \beta^*\hat{j}, d^*)$$

where θ^* is the required rotation, and $\alpha^*\hat{i} + \beta^*\hat{j}$ is the required translation for our problem.

For the clustering-based alignment as described above, we have to obtain the value of the θ such that d_θ, the density of the cluster of translation vectors, is maximized. However, it is found that conventional optimization routines are unable to find the correct alignment angle θ. This is because of two reasons:

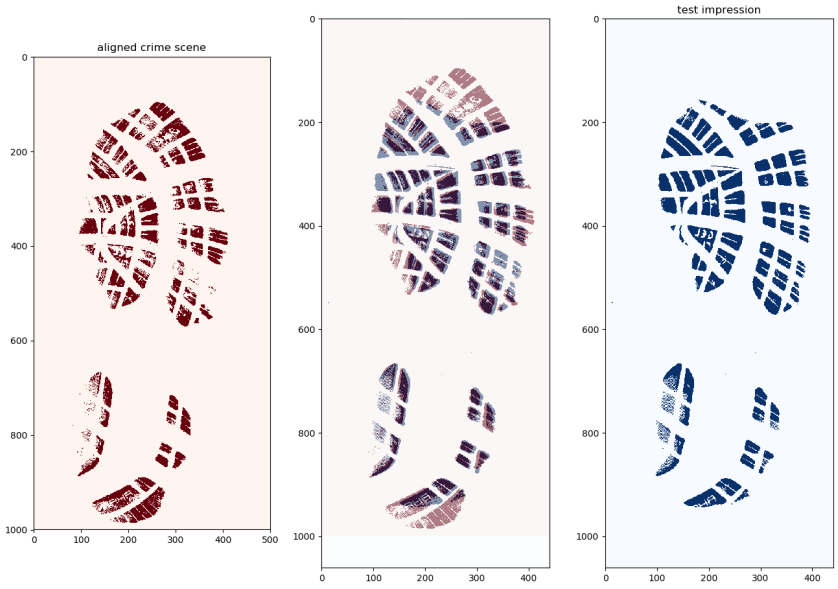

Figure 8: Alignment via clustering

1. **The optimization terminates at a local maximum instead of finding the global maximum:** If we plot the value of d_θ for each value of θ, we get a graph Figure 9:

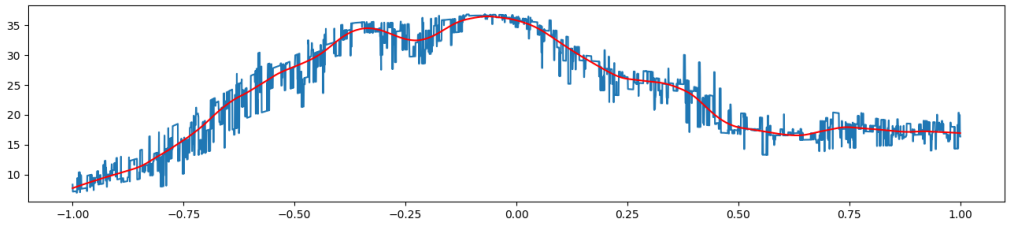

Figure 9: Blue indicates the actual curve, red is a best-fit smooth curve

2. **The densest cluster of translation vectors contains some duplicated vectors:** By *duplicated* we mean the following:
   - Let $\alpha^*\hat{i} + \beta^*\hat{j}$ be the optimum translation vector. Therefore the point (α^*, β^*) is the center of a dense collection of translation vectors $T^*(\theta) = \{t^*_{p,q}(\theta)\}$
   - Consider two members in $T^*(\theta)$, namely $t^*_{p,q_1}(\theta)$ and $t^*_{p,q_2}(\theta)$. Now, $t^*_{p,q_2}(\theta)$ is a duplicate of $t^*_{p,q_1}(\theta)$, because the point $p \in S_1$ **cannot correspond to two different points** q_1 and $q_2 \in S_2$. Hence only one of the two should be considered as part of the cluster. We can construct a similar case with two points from S_1 and one point from S_2.
   - Duplicate vectors of the above kind(s) are not eliminated while constructing our clusters, so it is possible that the optimum found in our process contains too many duplicates and is not actually the correct answer.

We can solve the first issue by brute force: iterate over all possible values of θ and pick the triplet with the maximum value of d_θ. For the second issue, we may need to define a *different* distance metric (instead of Euclidean) while clustering to account for removal of duplicates.

Due to the above reasons, the clustering-based alignment algorithm was not developed further. One might consider using distance metrics that account for duplicates while clustering. Since the translation vectors $t_{p,q}(\theta)$ are actually functions of θ, perhaps the distance metric can be defined between two functions of θ:

$$d(t_{p,q}, t_{p',q'}) = \min_{\theta \in [0,2\pi]} ||t_{p,q}(\theta) - t_{p',q'}(\theta)||_2$$

Graph-based Alignment

In the previous section we attempted to find the correspondence between points $p \in S_1$ and $q \in S_2$ by looking the distribution of the translation vectors over all possible rotations. Now, we model the correspondence as an undirected graph.

Given point-sets S_1 and S_2 as above, we define a graph $G(V, E)$ as follows:

- **Defining the set of vertices V:**
  - Each vertex v in the graph G corresponds to a tuple (p, q), where p is a point in S_1 and q is a point in S_2.
  - Therefore each vertex in the graph denotes a correspondence between a point in S_1 and a point in S_2.
  - If S_1 has M points and S_2 has N points, graph G has MN vertices i.e. $|V| = MN$.
- **Defining the set of edges E:** We draw edges in the graph G as follows:
  - Let v, v' be vertices in the graph G, such that v corresponds to a tuple (p, q) and v' corresponds to a tuple (p', q'). Here, $p, p' \in S_1; p \neq p'$, $q, q' \in S_2; q \neq q'$: the inequality prevents the duplication problem faced by the clustering approach.
  - Non we draw an edge between v and v' if:

    $$|d(p, p') - d(q, q')| \leq \epsilon$$

    for some $\epsilon > 0$, where d is the Euclidean distance metric.

Consider the two sets of points in Figure 10:

Since each vertex v in G maps a point $p \in S_1$ to a point $q \in S_2$, and the edges between vertices imply that the mappings (p, q) and (p', q') are *related*, we can now reduce the alignment problem to that of finding a *maximum clique* in the graph G. We define certain properties of cliques below (*Clique Problem*, n.d.):

- A complete subgraph H of a graph G is a clique.

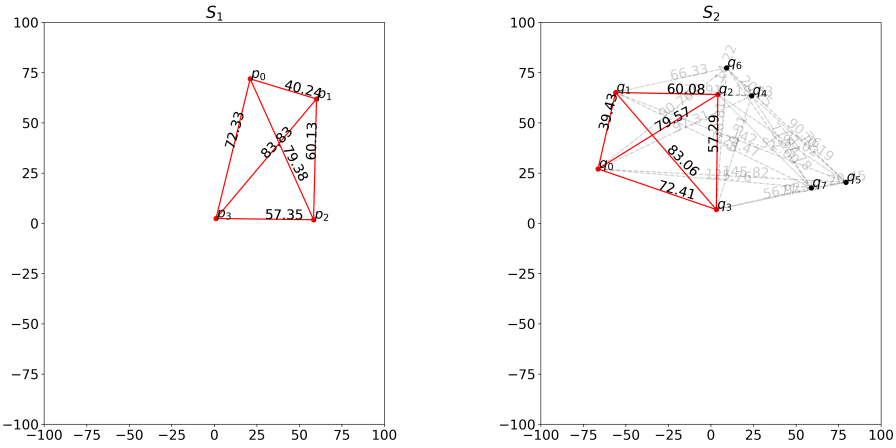

Figure 10: Distances marked on the lines joing the points

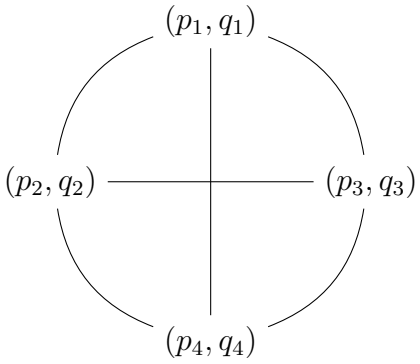

Figure 11: Clique in correspondence graph G

- A *maximal* clique H is a clique H that is not a subgraph of any other clique. It is possible to have multiple maximal cliques, as long as one does not contain the other.
- A *maximum* clique H is a maximal clique in the graph G such that there is no clique H' that is of larger size. It is possible to have multiple maximum cliques.

Therefore, given the point-sets S_1 and S_2 we now have to construct the graph G as above, and find a maximum clique in the graph. Once such a clique $H^*(V^*, E^*)$ is found, the set of vertices V^* of the clique are denoted as below:

$$V^* = \{v_1, v_2, ..v_n\} = \{(p_1, q_1), (p_2, q_2)....(p_n, q_n)\}$$

where $p_1...p_n \in S_1$ and $q_1..q_N \in S_2$. Thus the vertices of the clique H^* gives a list of corresponding points, which we can then use to find our required transformation T.

If we restrict the transformation T to just rotation and translation as earlier, we can use the Kabsch algorithm to obtain the required rotation θ^* and translation $\alpha^*\hat{i} + \beta^*\hat{j}$ to align the two shoeprint images. Alternatively, we can also use the corresponding points to fit a non-rigid transformation (like a polynomial or a spline transform) to account for a small range of shearing distortions.

It is well known that the *maximal* clique problem is in the **NP-Hard** class of problems; finding the solution to such problems is of exponential time complexity in the worst case. However, since our problem requires the discovery of just a **single maximum** clique, the computations can be sped up considerably in practice. Additionally, we can ensure that the graph G is sufficiently sparse, by using an appropriate value of ϵ while constructing the edges. This will also help reduce the computation time.

We find that the clique-based approach performs better than the clustering based approach, despite being slower in most cases.

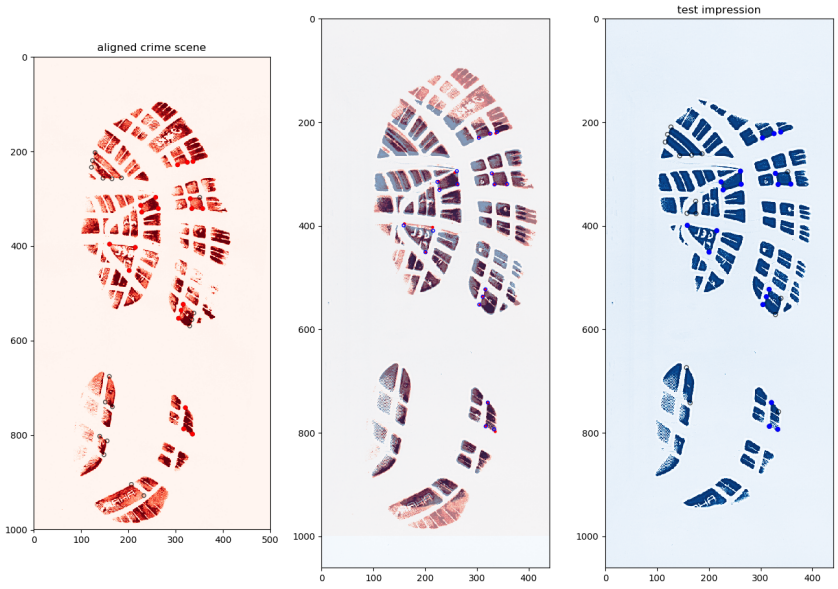

Figure 12: Graph-based alignment

Similarity Score Computation

Once the crime scene image and the database impression have been aligned, we can begin extracting features and computing similarity scores. Only the overlapping regions between the crime scene and the database impression are considered for this part of the process, ensuring that the amount of interfering information across both images is minimized.

As described in the introduction, most of the earlier methods to compute similarity scores involved the use of different kinds of transforms/filters and the concept of image moments. We study the basic concepts of the latter. (law Pawlak, 2006) was used as a primary reference.

Image moments and invariants

We consider the continuous case first: let $\Omega \subset \mathbb{R}^2$ denote the image plane, and let the image be a function $f(x, y) : \Omega \to \mathbb{R}$. Now we can define the moments of the function f:

1. **Geometric moments:**
   - The $(p, q)^{th}$ moment of f is defined as:

$$m_{p,q} = \int\int_{\Omega} x^p y^q f(x, y) dx dy$$

   - The set of geometric moments upto order N are : $\{m_{p,q} : 0 \leq p + q \leq N\}$
   - **Central moments** are geometric moments taken with a point $(\bar{x}, \bar{y})$ as the center of the image:

$$\mu_{p,q} = \int\int_{\Omega} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy$$

     where $\bar{x} = \frac{m_{1,0}}{m_{0,0}}$ and $\bar{y} = \frac{m_{0,1}}{m_{0,0}}$. Central moments are invariant to translation.
   - **Normalized central moments** can be obtained from central moments as follows:

$$\nu_{p,q} = \frac{\mu_{p,q}}{\mu_{0,0}^{1+\frac{(p+q)}{2}}}$$

Image moments are a method of dimensionality reduction; the entire information of the image is summarized in its moments. Hence we can compare relationships between two given images using relationships between their corresponding image moments. One of the earliest instances of image moments used as features was the seven **Hu moments** which can be constructed from the normalized central moments as follows:

$$\Psi_1 = \nu_{2,0} + \nu_{0,2}$$
$$\Psi_2 = (\nu_{2,0} - \nu_{0,2})^2 + 4\nu_{1,1}^2$$
$$\Psi_3 = (\nu_{3,0} - 3\nu_{1,2})^2 + (3\nu_{2,1} - \nu_{0,3})^2$$
$$\Psi_4 = (\nu_{3,0} + \nu_{1,2})^2 + (\nu_{2,1} + \nu_{0,3})^2$$

$$\Psi_5 = (\nu_{3,0} - 3\nu_{1,2})(\nu_{3,0} + \nu_{1,2})[(\nu_{3,0} + \nu_{1,2})^2 - 3(\nu_{2,1} + \nu_{0,3})^2] - (\nu_{0,3} - 3\nu_{2,1})(\nu_{0,3} + \nu_{2,1})[3(\nu_{3,0} + \nu_{1,2})^2 - (\nu_{2,1} + \nu_{0,3})^2] \quad (1)$$

$$\Psi_6 = (\nu_{2,0} - \nu_{0,2})[(\nu_{3,0} + \nu_{1,2})^2 - (\nu_{2,1} + \nu_{0,3})^2] + 4\nu_{1,1}(\nu_{3,0} + \nu_{1,2})(\nu_{2,1} + \nu_{0,3}) \quad (2)$$

$$\Psi_7 = (3\nu_{2,1} - \nu_{0,3})(\nu_{3,0} + \nu_{1,2})[(\nu_{3,0} + \nu_{1,2})^2 - 3(\nu_{2,1} + \nu_{0,3})^2] - (\nu_{3,0} - 3\nu_{1,2})(\nu_{0,3} + \nu_{2,1})[3(\nu_{3,0} + \nu_{1,2})^2 - (\nu_{2,1} + \nu_{0,3})^2] \quad (3)$$

Hu moments are invariant to rotation, translation, and scale; the lower order moments have some stability against noise in the image as well.

One can use geometric moments to represent a given image, however, these moments are highly correlated, and so there is redundancy in the information summarized by these moments. Also, reconstruction of the image from geometric moments is not accurate.

2. **Orthogonal moments:** One can use a system of 2D orthogonal polynomials to construct moments.
   - The $(p, q)^{th}$ moment of f is defined as:

$$\lambda_{p,q} = \int \int_{\Omega} V_{p,q}(x, y) f(x, y) w(x, y) dx dy$$

- Here $V_{p,q}$ belongs to a set of 2D orthogonal polynomials, and w is the corresponding weight function for that set. We construct a set of 2D orthogonal polynomials $V_{p,q}$ from sets of 1D orthogonal polynomials P_p, Q_q:

$$V_{p,q}(x,y) = P_p(x)Q_q(y)$$

- One of the classical orthogonal polynomials in one dimension are **Legendre Polynomials**, which are the set of polynomials defined by the following recurrence relation:

$$P_{n+1}(x) = \frac{2n+1}{n+1}xP_n(x) - \frac{n}{n+1}P_{n-1}(x); \quad P_0(x) = 1, P_1(x) = x$$

  With weight function $w(x) = 1$, and $\Omega = [-1, 1]$.
- Generalized 1D orthogonal polynomials: we have the class of **Gegenbauer** or **ultraspherical polynomials** with a parameter γ that allows for scaling. The recurrence relation is as follows:

$$G_{n+1}(x,\gamma) = 2\frac{(n+\gamma)}{(n+1)}xG_n(x,\gamma) - \frac{(n+2\gamma-1)}{(n+1)}G_{n-1}(x,\gamma)$$

  With weight function $w(x,\gamma) = (1-x^2)^{\gamma-\frac{1}{2}}$, $x \in [-1,1]$, and $\gamma > -\frac{1}{2}$. High values of γ lead to capture of more localized features in the image.
- Legendre Polynomials are a subset of Gegenbauer polynomials: when $\gamma = \frac{1}{2}$, the recurrence relation reduces to that of Legendre Polynomials.

One can construct invariants from these moments as well. Unlike geometric moments, orthogonal moments do not have high correlation, and error in the reconstruction of the image from these moments $\to 0$ as the order of moments considered $\to \infty$.

3. **Radial Orthogonal moments:** If we express the polynomials in polar form, we can obtain image moments which are invariant to rotation.
   - The class of **Zernike polynomials** can be expressed as such:

$$V_{p,q}(\rho cos\theta, \rho sin\theta) = R_{p,q}(\rho)e^{-iq\theta}$$

     Hence we can express the Zernike moments of an image as:

$$A_{p,q} = \int\int_{\Omega} V_{p,q}(\rho,\theta)f(\rho cos\theta, \rho sin\theta)\rho d\rho d\theta$$

$$A_{p,q} = \int \int_{\Omega} R_{p,q}(\rho) e^{-iq\theta} f(\rho cos\theta, \rho sin\theta) \rho d\rho d\theta$$

where Ω is the unit disk.

- We note that Zernike polynomials are orthogonal over the set Ω, and the magnitude of the moments $A_{p,q}$ are invariant to rotation.
- Given below is recurrence relation to generate the radial function $R_{p,q}$:

$$A_1 R_{p+2,q} = (A_2 \rho^2 + A_3) R_{p,q}(\rho) + A_4 R_{p-2,q}(\rho)$$

$$R_{q-2,q}(\rho) = 0; R_{q,q}(\rho) = \rho^q$$

where

$$A_1 = 2p(\frac{p+q}{2} + 1)(\frac{p-q}{2} + 1)$$

$$A_2 = 2p(p+1)(p+2)$$

$$A_3 = -q^2(p+1) - p(p+1)(p+2)$$

$$A_4 = -2(\frac{p+q}{2})(\frac{p-q}{2})(p+2)$$

Limitations of Moments and Invariants

Despite the many advantages of expressing images in terms of their moments and moment invariants, we see that the practical applications for our use case are limited due to the following factors:

- **Lack of representation of local relationships:** Since image moments (and the invariants constructed from them) are essentially a form of summary statistics of the image, the relationship between local features in an image is lost.
- **Discretization issues:**
  - Given that the images are usually arrays of pixel values, it is important to get accurate approximations of image moments in the discrete case.
  - Hence if a moment $\lambda_{p,q}$ is computed in the continuous form as below:

$$\lambda_{p,q} = \int \int_{\Omega} V_{p,q}(x,y) f(x,y) dx dy$$

the discrete approximation of $\lambda_{p,q}$ for an MxN pixel array is of this form:

$$\hat{\lambda_{p,q}} = \sum_{i=0}^{M} \sum_{j=0}^{N} h_{p,q} f(x_i, y_j)$$

where $h_{p,q}$ is a approximation of the integral about the pixel x_i, y_j:

$$h_{p,q} = \int \int_{nbd(x_i,y_j)} V_{p,q}(x,y) dxdy$$

- A simpler approximation of $\lambda_{p,q}$ can be taken as follows:

$$\hat{\lambda_{p,q}} = \Delta^2 \sum_{i=0}^{M} \sum_{j=0}^{N} V_{p,q}(x_i, y_j) f(x_i, y_j)$$

where Δ^2 is the area of the pixel, such that $V_{p,q}(x,y)$ can be considered constant over the pixel (x_i, y_j).

- Appropriate values of Δ and/or $h_{p,q}$ must be found in order to minimize error from discretization.

- **Sensitivity to noise and occlusion:** In tandem with the above issues, it is not known if the image moments described can provide distinguishing power in the case of partial images (as local relationships are not accounted for) or those with a considerable amount of noise.

- **Moment representation may not be unique:** It is seen that geometric moments do not uniquely represent an image, which means that the same feature vector of moments *can correspond to different images*. This is especially an issue when one-to-one correspondence between two images needs to be established.

- **Reconstruction accuracy:**

  - Though the reconstruction of images from their moments is exact in the limiting case, in practical cases, it is seen that moments of high order (≈ 50) are required to have visibly accurate reconstruction using Legendre polynomials. For Gegenbauer polynomials, it is possible to find N^*, the optimum number of moments for reconstructing an image, but this is usually obtained by cross-validation.
  - The reconstruction error can now be considered as due to three factors: discretization, noise in the image, and usage of finite

number of moments (limiting error). Depending on which of these is more significant for a given set of images, tradeoffs must be made in order to extract information with high distiguishing power.

- **Stability of invariants against unrelated transformations:**
  - As seen above, it is possible to construct image moments that are invariant to a given transformation (translation and rotation respectively). However, the constructed moments must also demonstrate some stability against *other* transformations on the image in order to have reliable distinguishing power. We see that this is not the case:
    - Legendre moments are invariant to translations, but sensitive to rotations in the image.
    - Zernike moments are invariant to rotations, but sensitive to translations in the image.
  - The issue of stability is compounded by the lack of local features, discretization and noise, which leads to the following question: given the feature vectors of image moments for two images, is it possible to identify *how much* a given transformation is affecting the distance between the two? Mathematically, given an image I, a transformation Φ, and moment-mapping $\Lambda : I \to \{\lambda_{p,q}\}$ which maps an image to a set of image moments, does

$$||I - \Phi(I)|| \to 0 \implies ||\Lambda(I) - \Lambda(\Phi(I))|| \to 0$$

    hold? In other words, *are the changes in the image accurately represented by the changes in the image moments*?
  - For example, if Zernike moments are used (i.e. Λ maps I to the set of its Zernike moments), and Φ denotes a small translation on the image I, is the distance between the feature vectors also small?
  - From the work of (Bruna & Mallat, 2013), it is seen that this is not the case: it is possible for image moments to have high sensitivity to certain kinds of transformations, such that even an unrelated image I' can be closer in terms of the image moments to I than $\Phi(I)$.

We see that an approach involving image moments and their invariants alone is not enough for the use case of shoeprint matching, especially in the case of

one-to-one comparisons. Though invariants do provide some distinguishing power, more flexible methods are required.

Using Localized Invariants

To provide flexibility in the process of feature extraction and similarity computation, a compromise is made while extracting image moments and their invariants:

- The area of influence for a given moment/invariant is *localized*, allowing reconstruction accuracy to be more stable on a per-patch basis. Recall that global image moments, due to being summary statistics, require many moments of higher order to accurately reconstruct the image.
- Local invariants represent basic image features. Extraction on a local level also mitigates the effect of noise.
- Once local invariants are extracted, the distribution of these can be used to model the more complex features of the image.

The advantage of using localized invariants is that it mitigates some of the instability seen earlier; now, only some parts of the extracted features may be affected due a particular transformation, occlusion, or noise. This enables us to use the information available on a given crime scene image more effectively.

Now the question arises as to how these localized invariants may be extracted. We first refer to *Active Basis Models*, from the work of (Wu, Si, Gong, & Zhu, 2010), which extracts edge features locally using a dictionary (or *basis*) of Gabor Filters.

Gabor Filters are defined as follows:

$$G(\lambda, \theta, \psi, \sigma, \gamma; x, y) = e^{-\frac{x'^2 + \gamma^2 y'^2}{\sigma^2}} e^{i(\frac{2\pi}{\lambda}x + \psi)}$$

Where:

- The center of the filter is assumed to be the origin,
- x, y refers to the point at which the response is being evaluated,

- $$x' = xcos\theta + ysin\theta$$

  ,

  $$y' = ycos\theta - xsin\theta$$

- σ, γ relate to the width (and height) of the Gaussian kernel that is used — this decides the local region of influence.
- θ, ψ relate to orientation and offset of the kernel — this function is not rotationally invariant, hence the dictionary of filters will need to contain many possible values of θ in order to get edges of different orientations.
- λ denotes the wavelength of this function — it is related to the thickness of the edge extracted.

Finally, the real and imaginary parts of this filters can be denoted using sine and cosine functions. The orthogonality of these functions allows to extract edges/stripes of both kinds (i.e. darker stripe between lighter areas, and vice versa).

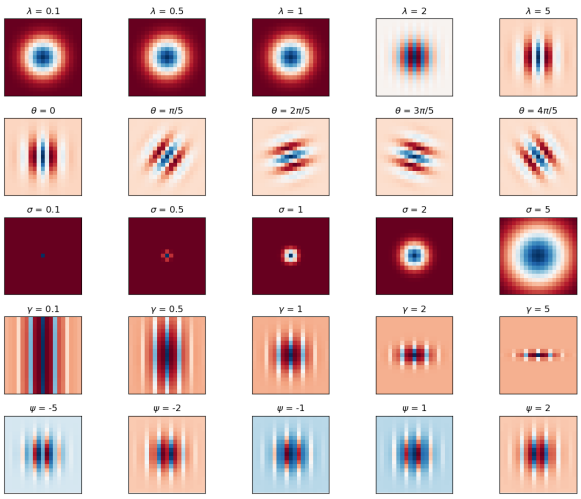

Figure 13: Gabor cosine kernels with respect to the changing parameters

Once the Gabor features have been extracted, the shape of the image is modelled as a "deformable image template" (as defined in (Wu et al., 2010)),

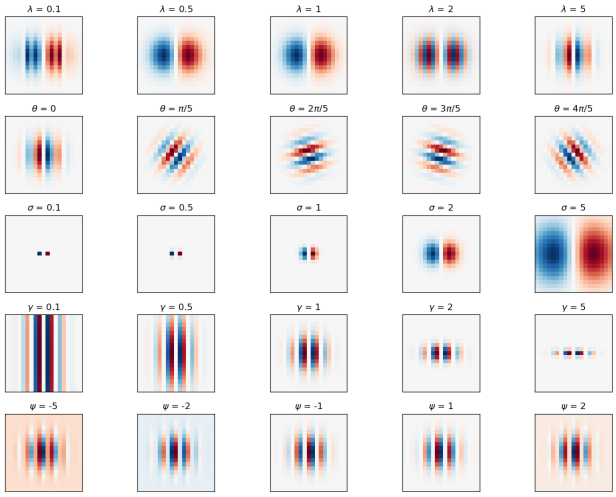

Figure 14: Gabor sine kernels with respect to the changing parameters

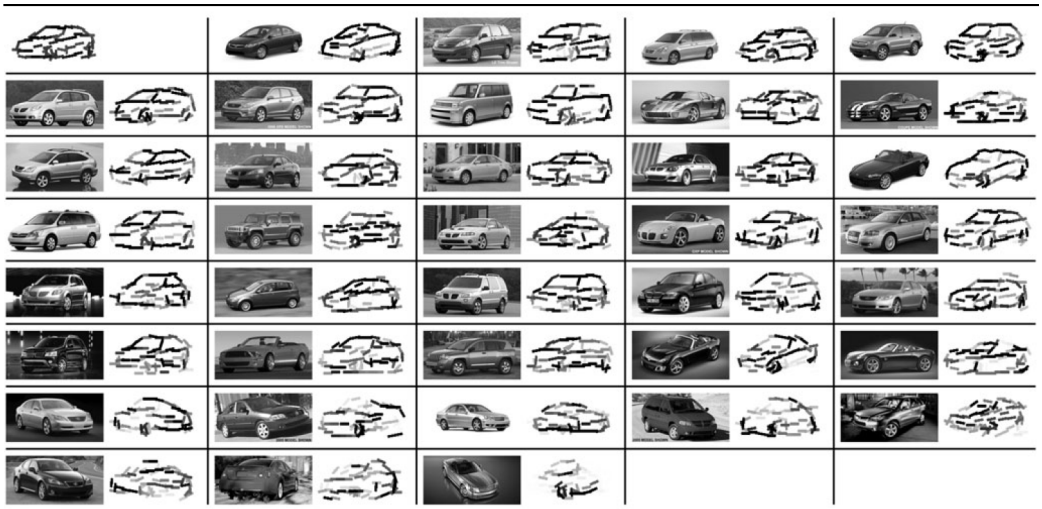

Figure 15: Deformable Image Template of a car, as given in (Wu et al., 2010)

meaning that local distortions between multiple versions of the same image can be handled without issue.

The problem of finding similarity between two images thus reduces to comparing the distance between their templates, and deciding whether the amount of deformation between the templates is acceptable. As said earlier, the flexibility of the image templates also means that occlusion does not affect the comparison process as much as it did for a purely invariant-based comparison.

However, a single level of abstraction (local features to image template) may not be sufficient to account for the various effects seen in crime scene impressions. For example, if there are some shapes from a shoe that are completely missing from a given crime scene impression, it may not be possible to describe this kind of occlusion or its effects on similarity.

In (Kortylewski & Vetter, 2016), we see that this detail is resolved by using *Compositional Active Basis Models*, which propose a tree-style hierarchy: non-leaf nodes correspond to different higher-level features in the image, and leaf nodes correspond to a dictionary of basic features. For the leaf nodes, a dictionary of Laplacian-of-Gaussian filters were used, in addition to the Gabor filters.

A Laplacian-of-Gaussian (LoG) filter centered at the origin is defined as follows:

$$L(x, y; \sigma) = \frac{-1}{\pi\sigma^4}(1 - \frac{x^2 + y^2}{2\sigma^2})(e^{-\frac{x^2+y^2}{2\sigma^2}})$$

Where σ denotes the radius of the 2D Gaussian kernel. Note that the LoG filter is rotation-invariant, and captures small, blob-like features from the image.

We now look at some image reconstructions using these dictionaries of filters in the active basis model:

From Figure 18 and Figure 19, we observe the following:

1. the active basis model constructs a version of the image that is more robust to effects from a crime scene, but
2. the reconstruction using the image template is affected by the choice of initial dictionary of filters

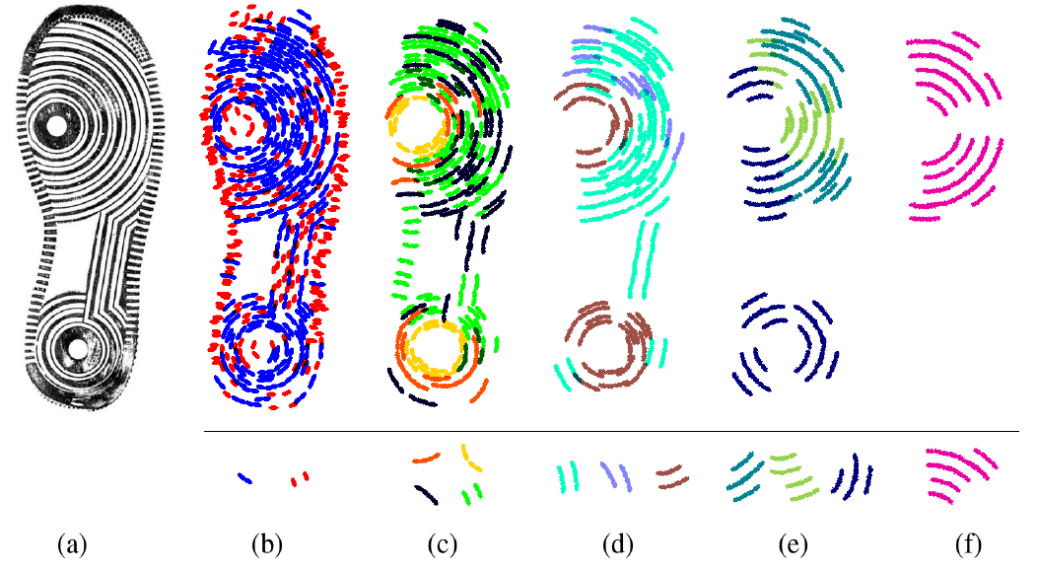

Figure 16: Hierarchical Compositional active basis models extract localized basic features at the leaf nodes as in (b), and the higher nodes combine to form more complex features from the shoe. Image taken from (Kortylewski & Vetter, 2016)

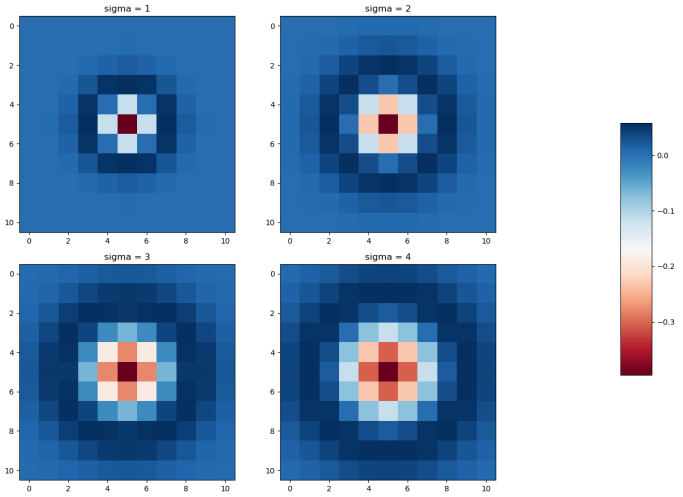

Figure 17: LoG kernels with respect to changing σ

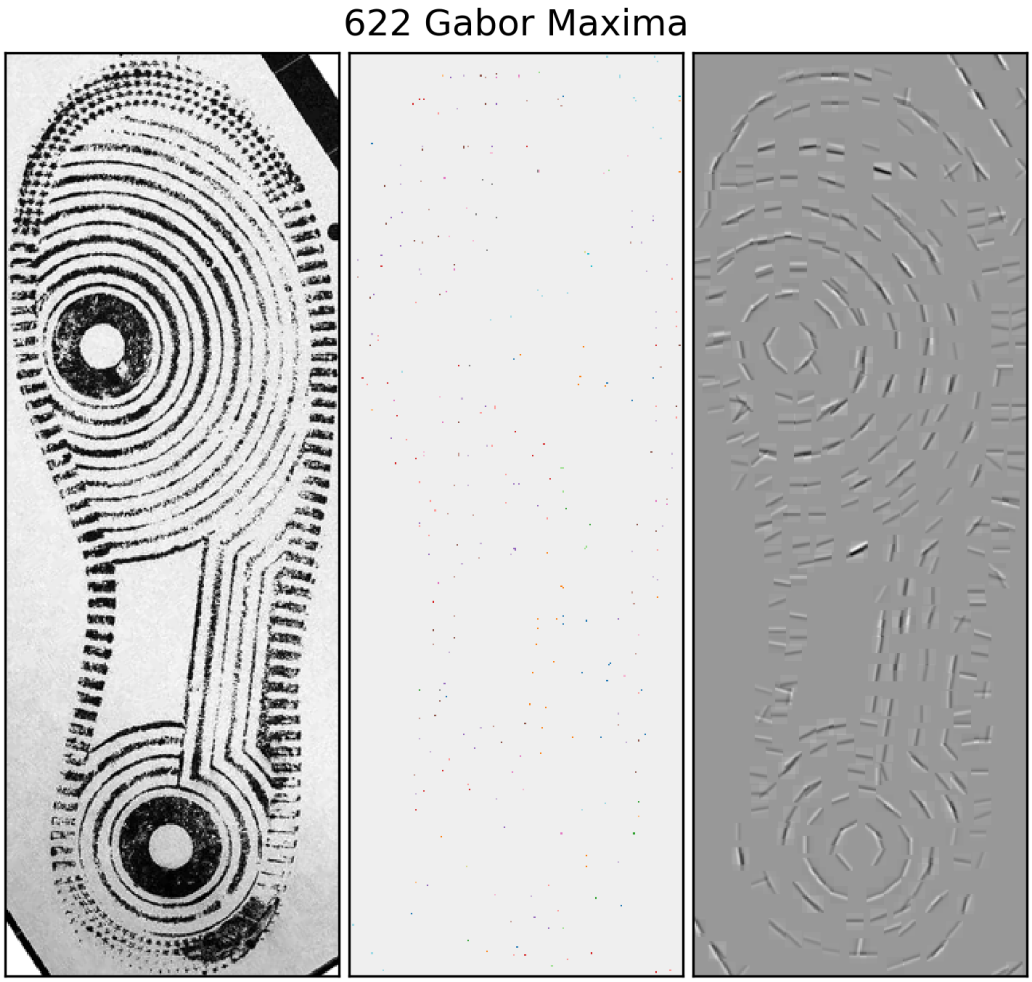

Figure 18: Shoeprint image from the dataset (Kortylewski et al., 2015) reconstructed with a basis model consisting of Gabor filters. Notice how the shapes have been broken up into line segments — these segments will later be part of some higher-order features. Also, the internal colour of the shapes is lost in this reconstruction.

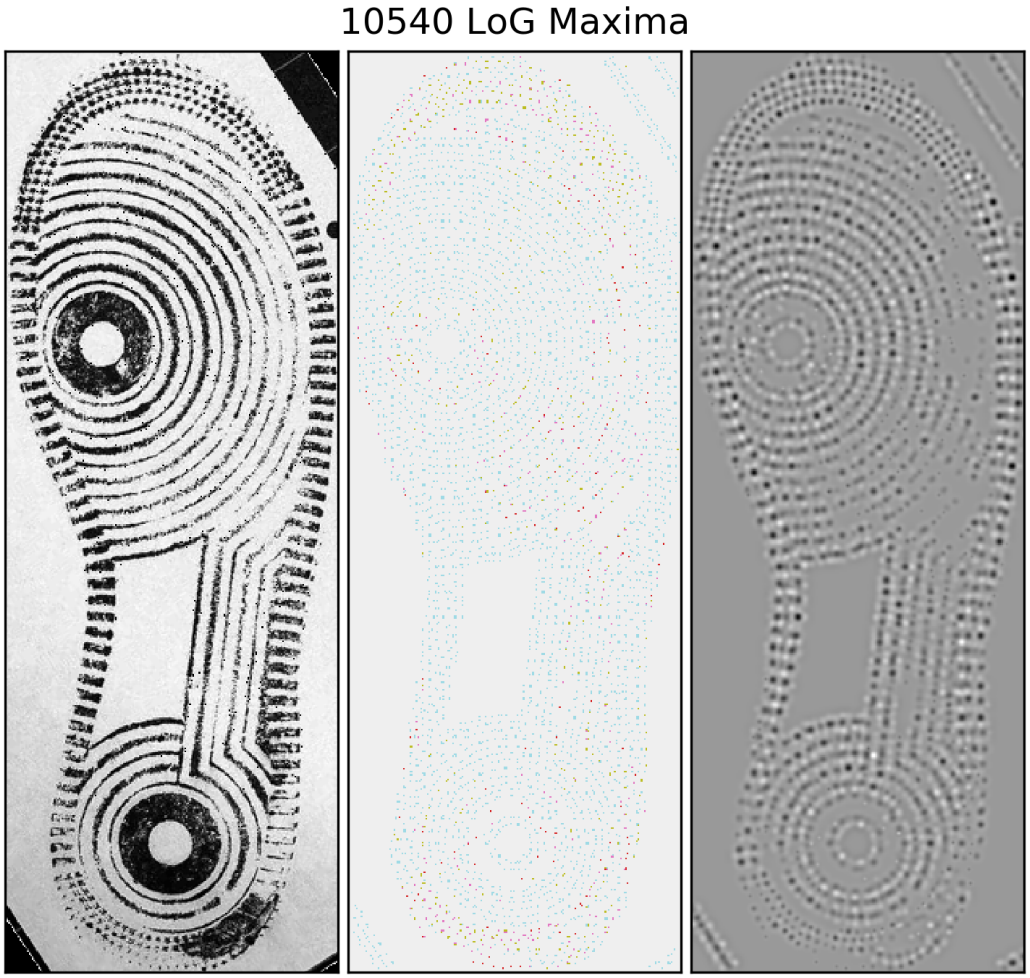

Figure 19: The same shoeprint image reconstructed with a basis model consisting of LoG filters. Notice how the shapes have been broken up into small blobs now — the edges are recorded implicitly unlike the previous image.

This leads to the following question: is there a complete dictionary of filters that can effectively describe the features specific a shoe-print image?

Feature Extraction using a Deep Neural Network

In the recent years, deep convolutional neural networks (CNNs) have gained widespread acclaim for their performance in image classification, conversion, captioning, etc. with many networks reporting close-to-human-level performance on such tasks.

Such a massive development was possible because of *automatic feature engineering*: unlike conventional methods of image processing (like we discussed earlier), the training of these networks allows one to specify the means (i.e. architecture of the model) and the end (ground truth data for training the model). The feature extraction is done automatically, with an aim to optimize the distinguishing power of the model.

Thus, using deep neural networks, we can attempt to obtain a more complete dictionary of filters that describe shoe-print images than conventional image processing filters. Additionally, the compositional nature of these networks ties in with the basis models; the structure and training of these networks may allow us to retain (and potentially improve on) the flexibility and generalization power from earlier.

Note that the filters of a neural network will only be obtained in an *implicit* manner. We can only view the weights of a trained model, as it may not be possible to describe these weights in a closed form.

Relation to Conventional Feature Extraction

It is interesting to note that the early layers of deep neural networks contain weights that are similar to conventional image processing filters. For example, the first layer of ResNet-50 (He, Zhang, Ren, & Sun, 2016) has weights (Figure 20) that have a similar distribution to the values from a Gabor Filter (Figure 21).

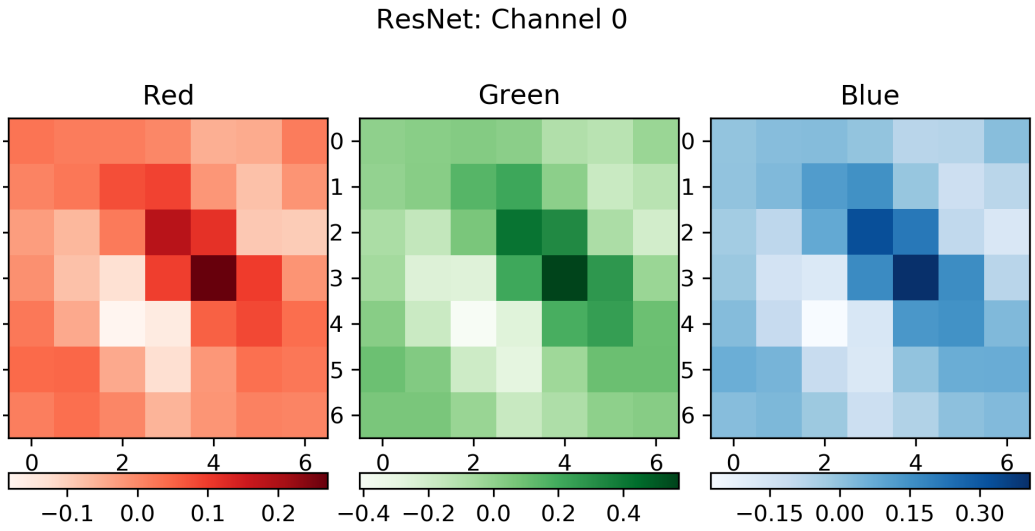

Figure 20: Weights for Channel 0 of ResNet-50

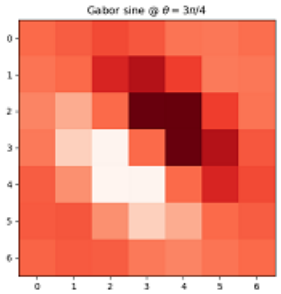

Figure 21: Gabor sine kernel

However, we see that the dictionary of filters learned by ResNet-50 (a neural network trained on kinds of images) does not consist of just Gabor filters at different angles; this shows that there are generalized basic features (not just edges) that can be extracted from any natural image.

Training a DNN

However, it must be noted that deep neural networks are prohibitively expensive to train:

- a large amount of (labelled) training data is required,
- the complexity of the network means training will take more time
- hyper-parameter tuning for producing an optimum result is another time sink.

In the context of shoe-prints, the amount of data available to train with (i.e. test impressions and crime scene images) is very limited. The dataset from (Kortylewski et al., 2015) has around 1000 test impressions, and is the only major shoe-print dataset available publicly.

Since a large training dataset is not available, the concept of *transfer learning* comes into play. Specifically, we can use pre-trained network models as *fixed feature extractors*: the activations from the intermediate layers of these deep neural networks can be used as features with which we can compare two images.

Experiments

We wish to see the scope of pre-trained models for feature extraction and one-to-one similarity computation in the shoe-print domain. We shall compare pairs of shoe-print images in a manner similar to that described in (Kong et al., 2017):

1. extract intermediate features from a pre-trained model,
2. view the distinguishing power of individual channels, in terms of normalized cross correlation (NCC) and phase-only correlation (POC),

and

3. find the averaged unweighted multi-channel normalized cross correlation (MCNCC) and multi-channel phase-only correlation (MCPOC) score.

We will use a subset of the data available in (Kortylewski et al., 2015), comparing 300 pairs of known matches and 600 pairs of non-matches. Since we wish to view only the one-to-one comparison, the Receiver Operating Characteristic (ROC) curve, and the area under it (AUC) will be used as the evaluation metrics.

ResNet

We use the ResNet-50 model (He et al., 2016), and extract activations from layers `Resnet-1bx` and `Resnet-2bx`, as described in (Kong et al., 2017).

After obtaining the activations, we find the per-channel NCC and POC and use them as similarity scores for computing the ROC curve.

SqueezeNet

We use the SqueezeNet1.0 model (Iandola et al., 2016), and extract activations after the second`Fire` layer.

After obtaining the activations, we find the per-channel NCC and POC and use them as similarity scores for computing the ROC curve.

VGG-Net

We use the VGG-19 model, and extract activations after the tenth layer.

After obtaining the activations, we find the per-channel NCC and POC and use them as similarity scores for computing the ROC curve.

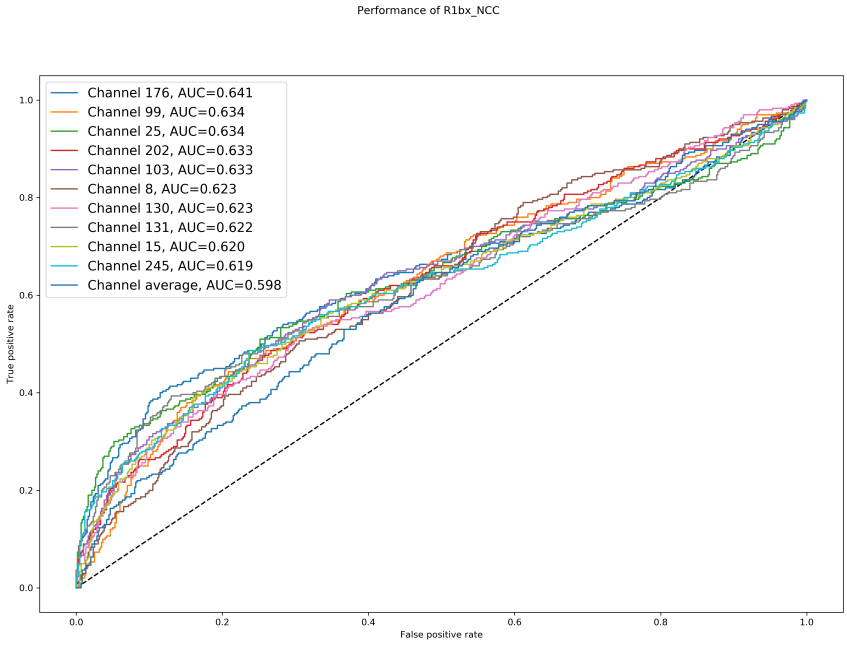

Figure 22: `Resnet-1bx` per-channel NCC.

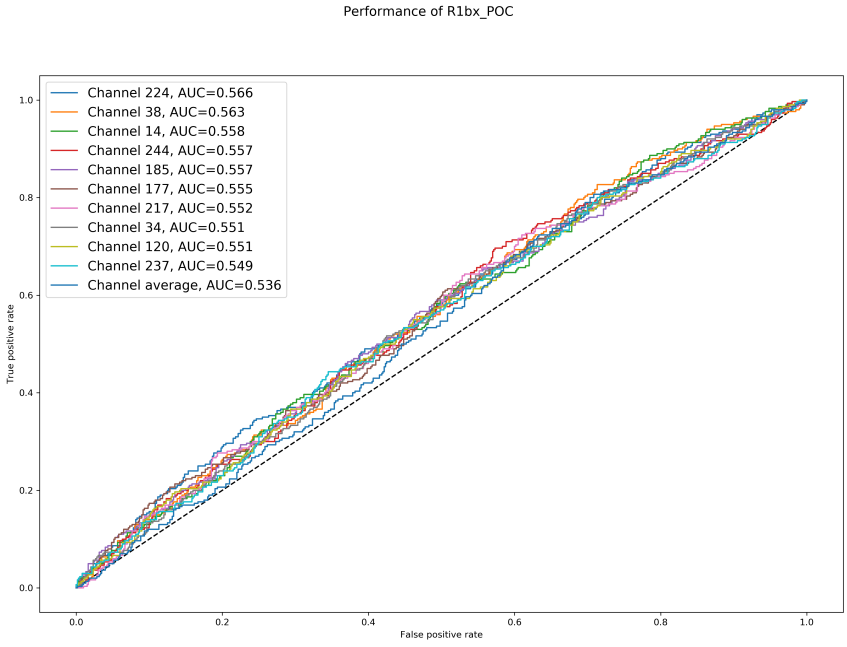

Figure 23: `Resnet-1bx` per-channel POC.

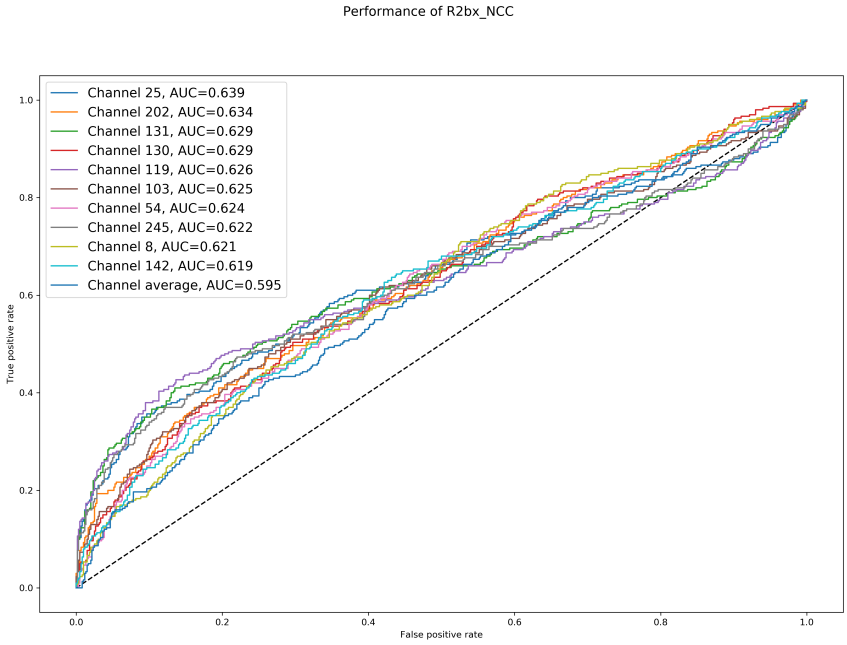

Figure 24: `Resnet-2bx` per-channel NCC.

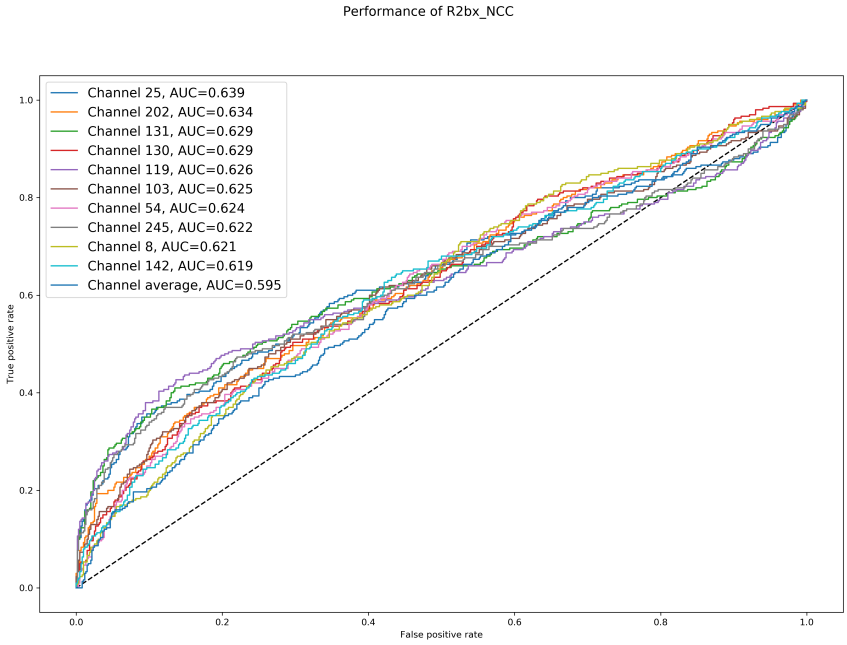

Figure 25: `Resnet-2bx` per-channel POC.

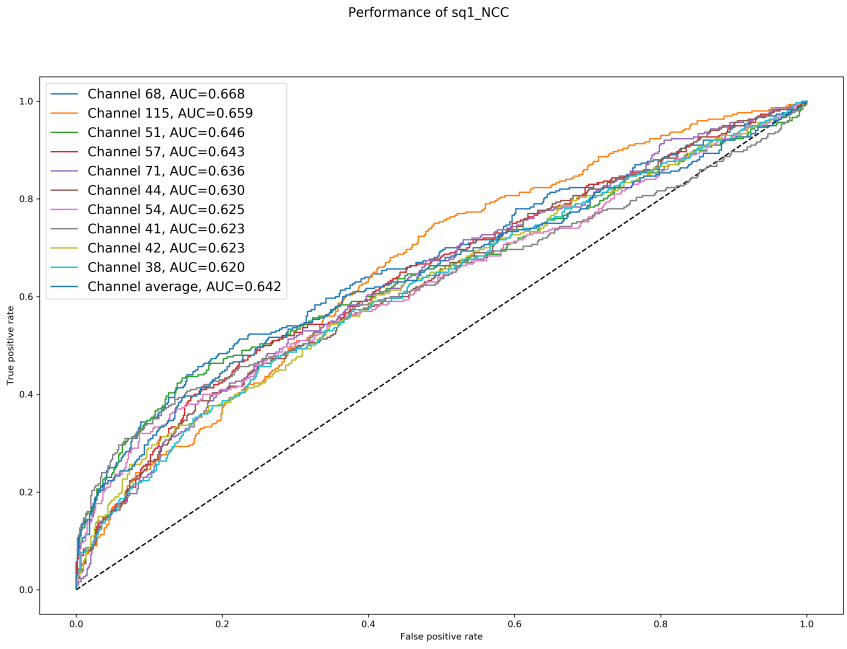

Figure 26: `SqueezeNet1.0` per-channel NCC.

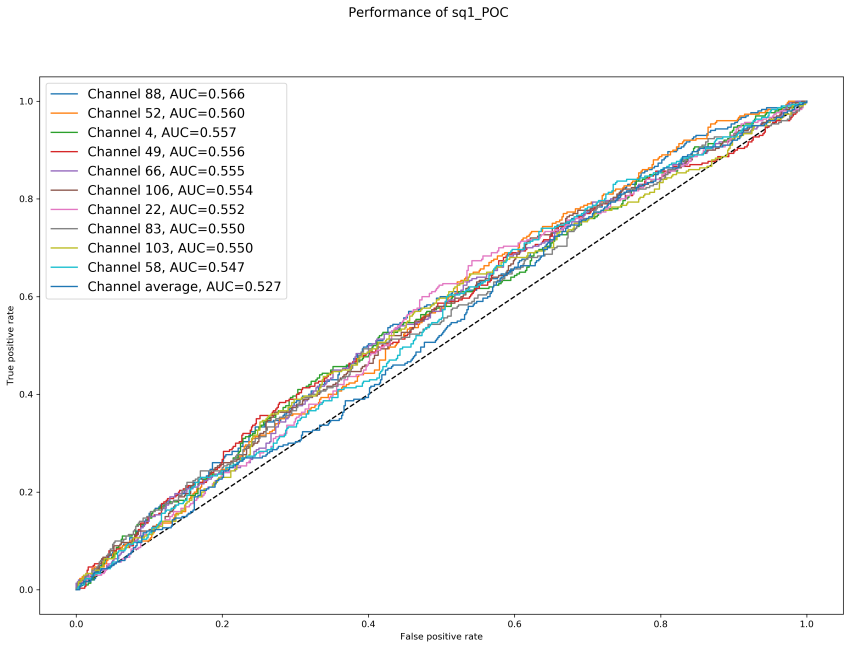

Figure 27: `SqueezeNet1.0` per-channel POC.

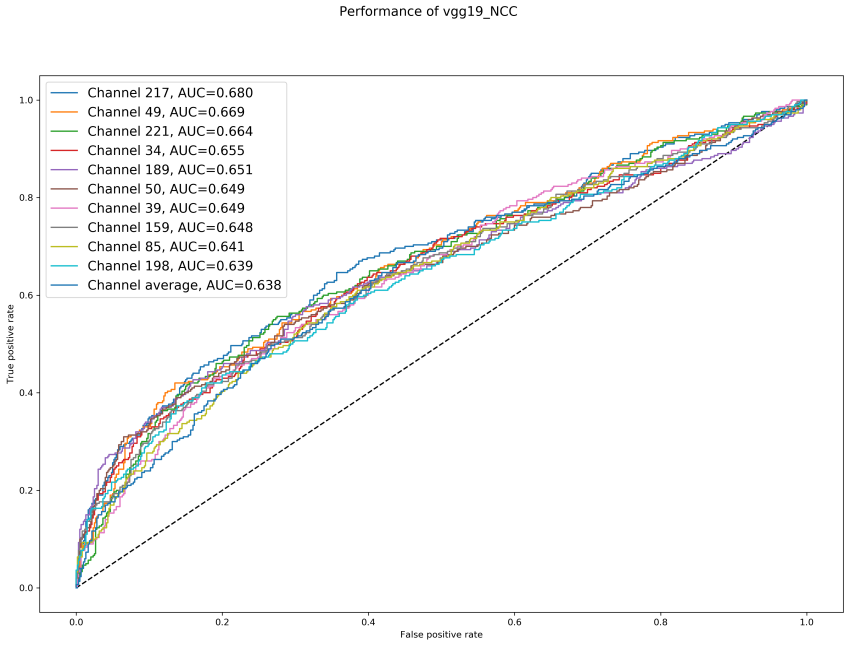

Figure 28: `VGG-19` per-channel NCC.

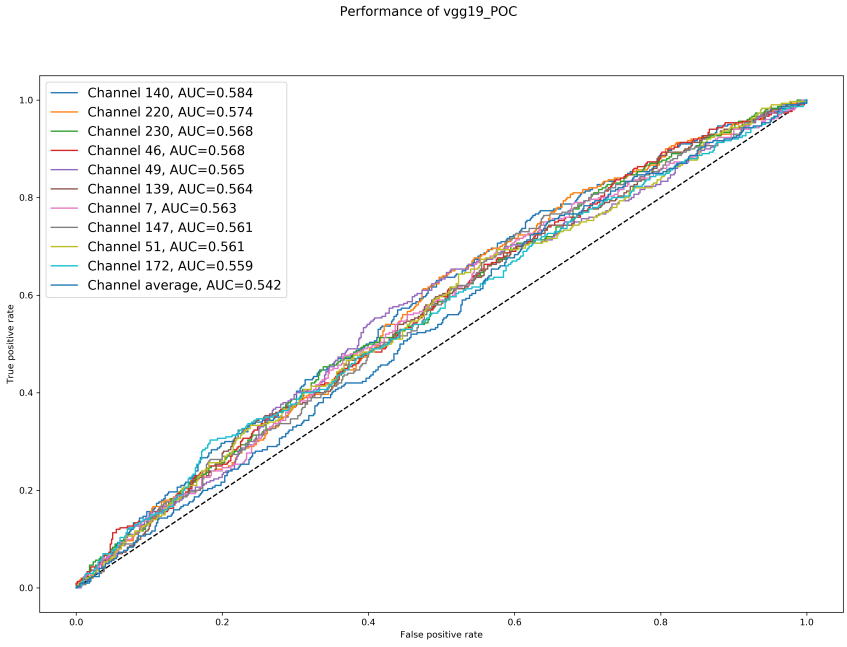

Figure 29: `VGG-19` per-channel POC.

Conclusions and Future Work

In this work, we studied the potential applications of machine learning to analysis of forensic footwear evidence.

- We considered the contexts of database retrieval, one-to-one comparison, and how a potential method would have to account for the differences between the two,
- We implemented a process to manually mark-up crime-scene images before starting the comparison, as complete automation was not possible
- We implemented a variant of the FAST algorithm to suit the use case of high-precision corner extraction from shoe-prints
- We implemented a method of image alignment using graph theory and points marked on the image,
- We studied the conventional methods of feature extraction from images, some of their limitations, and arrived at the doorstep of deep neural networks,
- We studied the requirements for training a deep learning model from scratch, and tested the potential of transfer learning for our use case: the transferred models produce a reasonable ROC even with no additional training.

While deep neural networks do have potential for robust feature extraction and comparison, the amount of data at hand (especially crime-scene images) is limited. Further data must be collected for testing and developing more networks: one can use the intermediate features as a base, add a few more layers and train the whole network afresh.

However, the study of this field has led to many interesting questions regarding the use of feature extraction: for example, would it be possible to use features from a neural network to align a pair of shoe-print image? There can be developments aimed at making the feature from a neural network more understandable by humans, as seen in (Kortylewski & Vetter, 2016). One can also explore if it would be possible to train a neural network to automatically align a given pair of images, thus eliminating the need for manual mark-up! The possibilities are endless.

References

Bouridane, A., Alexander, A., Nibouche, M., & Crookes, D. (2000). Application of fractals to the detection and classification of shoeprints. In *Image processing, 2000. proceedings. 2000 international conference on* (Vol. 1, pp. 474–477).

Bruna, J., & Mallat, S. (2013). Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1872–1886.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*(6), 679–698.

Clique problem. (n.d.). Retrieved 2018-10-05, from `https://en.wikipedia.org/wiki/Clique_problem`

Duda, R. O., & Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, *15*(1), 11–15.

Geradts, Z., & Keijzer, J. (1996). The image-database rebezo for shoeprints with developments on automatic classification of shoe outsole designs. *Forensic Science International*, *82*(1), 21–31.

Gueham, M., Bouridane, A., Crookes, D., & Nibouche, O. (2008). Automatic recognition of shoeprints using fourier-mellin transform. In *Adaptive hardware and systems, 2008. ahs'08. nasa/esa conference on* (pp. 487–491).

Harris corner detector. (n.d.). Retrieved 2018-10-05, from `http://scikit-image.org/docs/dev/api/skimage.feature.html#skimage.feature.corner_harris`

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*.

Kong, B., Ramanan, D., & Fowlkes, C. (2017). Cross-domain forensic shoeprint matching. In *British machine vision conference (bmvc)*.

Kortylewski, A., Albrecht, T., & Vetter, T. (2015). Unsupervised footwear impression analysis and retrieval from crime scene data. In *Computer vision - accv 2014 workshops* (pp. 644–658). Springer International

Publishing.

Kortylewski, A., & Vetter, T. (2016). Probabilistic compositional active basis models for robust pattern recognition. In *Bmvc.*

law Pawlak, M. (2006). Image analysis by moments: reconstruction and computational aspects. *Oficyna Wydawnicza Politechniki Wrocławskiej.*

Patil, P. M., Deshmukh, M. P., & Kulkarni, J. V. (2012). Investigation of shoeprints using radon transform with reduced computational complexity. *Journal of Pattern Recognition Research*, *7*, 80–89.

Patil, P. M., & Kulkarni, J. V. (2009). Rotation and intensity invariant shoeprint matching using gabor transform with application to forensic science. *Pattern Recognition*, *42*(7), 1308–1317.

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In *European conference on computer vision* (pp. 430–443).

Sawyer, N. E., & Monckton, C. W. (1995, May). "shoe-fit"-a computerised shoe print database. In *European convention on security and detection, 1995.* (p. 86-89). doi: 10.1049/cp:19950475

Shi-tomasi corner detector. (n.d.). Retrieved 2018-10-05, from `http://scikit-image.org/docs/dev/api/skimage.feature.html#corner-shi-tomasi`

Wu, Y. N., Si, Z., Gong, H., & Zhu, S.-C. (2010, Nov 01). Learning active basis model for object detection and recognition. *International Journal of Computer Vision*, *90*(2), 198–235. Retrieved from `https://doi.org/10.1007/s11263-009-0287-0` doi: 10.1007/s11263-009-0287-0