SEARCH OPTIMIZATION THROUGH JPEG QUANTIZATION TABLES

USING A DECISION TREE LEARNING APPROACH

SHARON A. GIESKE
6167667

Master Reseach Project #2
Credits: 6 EC

Master Program System and Network Engineering

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
dr. ing. Z.J.M.H. Geradts

Netherlands Forensic Institute
Laan van Ypenburg 6
2497 GB DEN HAAG

July 11th, 2014

ABSTRACT

This thesis describes the research conducted for optimizing search through JPEG quantization tables. Digital images are frequently found in forensic investigation. In these cases it can be very important to identify the source device of these images. Camera identification can be performed with the use of JPEG quantization tables that are present in the JPEG image. The tables can be matched against all known tables that are retrieved from images from different camera makes and models. This method can be costly in time and an optimization in the search through these JPEG quantization tables is desired.

In this research, a decision tree learning algorithm is applied in order to create decision tree models for the prediction of the camera make as well as the camera model of an image. The performance of the decision tree model is compared to two implementations of prediction models that use databases containing the hashes of the JPEG quantization tables.

The decision tree model performs satisfactory and gains an F_2-score of 89% for the prediction of the camera make and an F_2-score 80% for the prediction of the camera model. It gains high scores on recall, which is important for forensic investigations in order to retrieve more incriminating evidence, as well as precision, which is important for decreasing search space. Because the decision tree model creates a one-to-one mapping of JPEG quantization tables to a single camera make or model, a trade-off in F_2-scores between classes is found when the JPEG quantization tables are retrieved from images with a different camera make or model. The decision tree model performs better for the prediction of the camera make because camera models of the same make often use the same JPEG quantization tables.

The decision tree model also performs satisfactory in comparison with the database prediction models. The database prediction models score lower on precision, from which can be concluded that an optimization in search is made with the decision tree model. The decision tree model also identifies 50 important parameters from 128 values in the JPEG quantization tables during feature selection. Because not all values in these tables are used, in contrast to the hash database models, the decision tree model is also more flexible when small changes in the JPEG quantization tables occur.

Overall, the decision tree learning algorithm has a good performance for optimizing search through JPEG quantization tables.

CONTENTS

1 INTRODUCTION

This chapter gives an introduction for the research on search optimization for JPEG quantization tables. The motivation for this research is explained and the research question on which is focused is stated. This chapter also describes related work for this research.

1.1 MOTIVATION

Taking pictures is very easy and popular in this digital age. The demand for digital cameras was forecast to be 86 million units for 2013 by Futuresource Consulting[6]. And even though a decline in market share is present for digital cameras, the worldwide sales in 2013 of smartphones to end users increased with 42.3% from 2012 and totalled a sale of 968 million units in 2013, as reported by Gartner[7]. Due to this proliferation of smartphones (nowadays all equipped with a camera function) the total number of digital images taken each year is very high. Social media sites which have photo upload functions, such as Facebook and Instagram, report significantly huge numbers on the total upload of images. Facebook alone reported in a white paper [1] that more than 250 billion photos are uploaded to their site, with on average a total upload of more than 350 million photos every day. Statistics on Instagram[1] show a total of 20 billion photos shared on Instagram.

Due to this popularity, digital images are often recovered in forensic investigation. For example, in child pornography cases many digital images are present and are important evidence for the investigation. In such a case it can be very important to identify the origin of images to a specific camera or identify images that come from a common source. This can be achieved by uncovering traces on pictures that are distinguishable for camera models. One of these traces is the JPEG quantization table, which is specified as a set of 8 × 8 (integer) values. Separate quantization tables are employed for luminance and chrominance data, where some implementations include two chrominance quantization tables; one for chrominance-red and one for chrominance-blue.

In order to match JPEG quantization tables a comparison between 128 values, or 192 when two chrominance quantization tables are present, is made. With over a dozen different camera brands, each developing different models over the years, the number of camera models (and consequently the number of JPEG quantization tables) to be matched against is significantly high. The matching of large databases of images against these camera models will be time costly as for every match process 128 or more integer comparisons are made. This matching process needs to be minimized since time is often limited in forensic investigations. This research will focus on optimizing search through the image databases regarding JPEG quantization tables.

[1] http://instagram.com/press/ accessed 03-06-2014

As camera identification can be seen as a pattern recognition problem[10], machine learning algorithms can be applied to create predictive models for camera identification. These algorithms are able to handle large datasets and can be used to optimize search for patterns in datasets. A machine learning algorithm that is easy to interpret and to implement in other search systems is the decision tree learning algorithm. In this research a decision tree learning algorithm is applied for optimizing search through the image databases regarding JPEG quantization tables.

1.2 FOCUS OF RESEARCH

The research question on which is focused is set as: *'Can searching through JPEG quantization tables be optimized with the use of decision tree learning?*

In order to answer the research question, this research will focus on the following subquestions:

1. Can identifiable parameters be found in JPEG quantization tables?
2. What is the performance of decision tree learning with JPEG quantization tables?

In the following chapters the answers to these questions will be given.

1.3 RELATED WORK

Research on digital image forensics is a growing field. It focuses on two main interests, namely source identification and forgery detection. Van Lanh et al. [12] created a survey on digital camera forensics, which describes several techniques in these two fields. Their survey shows the use of intrinsic features of camera hardware and software for camera identification and concludes that hardware features give more reliable and better result. To distinguish between cameras of the same model imperfections of camera the use of hardware features seems to be the best method. Methods for forgery detection also rely on hardware-dependent characteristics but show lower accuracy rates compared to camera identification methods. In another survey, Weiqi et al. [10] describe methods for passive technology for digital image forensics. They state that in most cases passive forensics can be converted to a problem of pattern recognition.

In forgery detection, methods to identify JPEG quantization tables are often used. In research by Kornblum[9] quantization tables used by several image software are identified. A software library called Calvin is developed to identify those images who cannot be guaranteed to have been created by a real camera. Research by Farid[5] shows a technique for detecting tampering in low-quality JPEG images by identifying a cumulative effect of quantization.

JPEG quantization tables can also be used for source identification. Farid has performed research[3][4] on source identification with the use of JPEG quantization tables. This research states that a sort of camera signature is embedded within each JPEG image due to the used JPEG quantization tables since they differ between manufacturers. Although the JPEG quantization is not perfectly unique, the majority of cases where different camera models share the same JPEG quantization tables is for cameras from the same manufacturer. It states that (the use of JPEG quantization tables) "*is* *reasonably effective at narrowing the source of an image to a single camera make and model or to a small set of possible cameras.*" (p. 3)

There exist several projects where JPEG quantization tables are used as camera signatures. For example, the JPEGsnoop[2] project reports a huge amount of information to expose hidden information in images. Another project is the (discontinued) commercial FourMatch[3], which was focused on forgery detection. These projects compare the camera signature found for an image with a database of camera signatures to identify the camera make and model. These projects are not focused on matching large sets of images against a large camera database. In contrast, this research hopes to contribute by creating a decision tree model that can be used to decrease the search space for large datasets and which can easily be combined further with other (more accurate) source identification techniques.

[2] http://www.impulseadventure.com/photo/jpeg-snoop.html

[3] http://www.fourandsix.com/fourmatch

2 BACKGROUND

In the following sections several techniques are described that are used in this research. This can give the reader adequate background information to fully understand the implementation.

2.1 JPEG QUANTIZATION TABLES

JPEG quantization tables are created during the JPEG compression phase. In this section a description is given of the JPEG compression method.

The storing of a raw image format is often undesirable since this requires much storage space. In order to reduce the storage space for an image, compression methods are used to create an appropriate trade-off between file size and image quality. JPEG is a commonly used method for lossy compression of digital images. This compression technique is based on the discrete cosine transform (DCT) and is lossy because the original image information is lost and cannot be restored, possibly affecting image quality.

The JPEG compression is composed out of several steps which are depicted in Figure 1. First, a color space conversion is made from the RGB domain to the YC_bC_r domain. The YC_bC_r domain uses luminance, chrominance blue and chrominance red. The luminance describes the brightness of the pixel while the chrominance carries information about its hue. This color space conversion is chosen because people are significantly more sensitive for changes in luminance than chrominance and as a result the chrominance channels can be down sampled easily with almost no visual effect.

In the second step the image is split into blocks of 8×8 pixels. For each block, each of the Y, C_b and C_r data undergoes the DCT. The DCT transforms a signal or image from the spatial domain to the frequency domain. Thirdly, the amplitudes of the frequency components are quantized. Because human vision is more sensitive to variations over large areas than to strength of high-frequency brightness variations, the magnitudes of the high-frequency components are stored with a lower accuracy than the low-frequency components. Each component in the frequency domain is divided by a constant for that component, and then rounded to the nearest integer. These constants are stored in quantization tables. Seperate quantization tables for both the luminance as the chrominance domain are used, where chrominance blue and chrominance red are often combined to one table. The elements in the quantization table thus control the compression ratio.

In the last step of JPEG compression, entropy encoding is used. The image components are arranged in a "zigzag" order after which the compression method employs a run-length encoding (RLE) algorithm. This algorithm stores sequences of data as a single data value and count, rather than as the original run. It then inserts length coding zeros and uses Huffman coding.

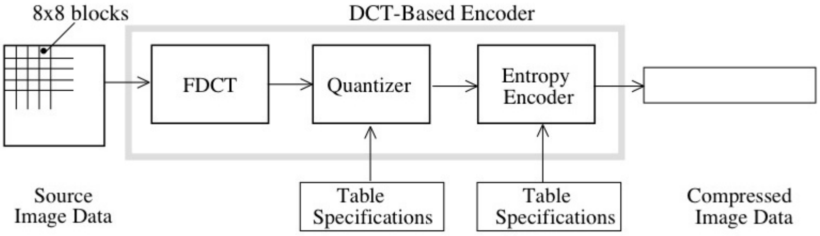

Figure 1: JPEG compression steps

Source: Fig. 1. DCT-Based Encoder Processing Steps. From: Wallace, G. K. (1992). *The JPEG still picture compression standard.* Consumer Electronics, IEEE Transactions on, 38(1), xviii-xxxiv.[13]

The JPEG image is saved with a JPEG File Interchange Format (JFIF) which contains several headers. These headers consist of markers and their associated information and are used for compatibility. For example, JPEG does not define the color space that is to be used for the image. JFIF has a marker in which the color space in use can be defined. The JPEG quantization tables are also stored in these headers.

2.2 DECISION TREE LEARNING

Decision tree learning is a supervised machine learning algorithm. It aims to construct the best decision tree model from class-labelled training. It is a predictive modelling approach as its goal is to create models to predict a value of a target based on input variables which are the target attributes.

A decision tree has a flow-chart-like structure where internal nodes denote a test on attributes and the leaf nodes hold a class label. Each branch corresponds to an attribute value. The decision tree holds a mapping of observations in attributes of a target to a class. The decision tree learning algorithm aims to create the best 'splits' (i.e. the attribute tests) by finding patterns in the set of attributes. These splits are learned by recursive partitioning: the algorithm splits the data set into subsets with specific attribute values (i.e. patterns) and repeats this for the subsets until the subset is correctly partitioned to belong to the same target value, or when splitting no longer adds value to predictions.

The decision tree learning algorithm has many advantages. The decision tree is easy to interpret since it is comprised out of rules of boolean logic. In contrast to other machine learning algorithms, it requires little data preparation such as data normalization. For these reasons, a decision tree model can be implemented in the use of other search systems, for example in the use of databases queries. The decision tree learning algorithm is also able to handle both numerical and categorical data. This is useful since the quantization tables used in this research comprise of numerical values. In addition, the cost for predicting data is logarithmic in the number of data points used to train the tree.

However, the decision tree learning algorithm also has some disadvantages. The decision tree learning algorithm can create over-complex trees as it is a greedy algorithm: it creates locally optimal solutions at the splits of the decision tree that approximate a global optimal solution. This can result into overfitting of the data as it does not generalize well from the training data. Decision trees can therefore be somewhat unstable as small variations in the data can result into different decision trees.

In this research the decision tree learning algorithm is chosen mainly because it is easy to interpret and little data preparation is needed. Another reason is that its decision tree can be implemented in other search systems.

3 EXPERIMENTAL SETUP

In this chapter an overview is given of the approach to use decision tree learning for optimizing search through JPEG quantization tables. It describes the several steps and their implementations in this research.

3.1 DATASET

The dataset that is used in this research consists of 45,666 images. These images are retrieved from the Dresden Image Database [8] and an image database from the Netherlands Forensic Institute. This dataset has images from 19 different camera makes and a total of 41 different camera models. The camera make and models are listed in Table 3 together with the number of pictures made with these cameras. Next to regular digital cameras there are also images from other types of cameras included in the dataset, such as images taken by smartphones (e.g. Blackberry), a webcam (Logitech), scanners (Epson) and a Playstation device (PS Vita).

3.2 APPROACH

In this section the approach is given for the prediction of the camera make as well as the camera model based on the JPEG quantization tables. First, all JPEG quantization tables are extracted from the images and stored with the corresponding camera make or model label. Next, these tables are converted to simple feature sets and several extra features are added. On these feature sets a feature selection is performed to retrieve the most important features. The set of important features and corresponding labels is split into a training set and a test set. The training set is used as input for in the decision tree classifier which returns a decision tree model. This model is then used on the test set in order to evaluate its performance. In addition, two different prediction models, which use a database in which JPEG quantization tables, are hashed and stored with their labels, are created. In order to give a good view on the performance of the decision tree classifier, its performance is compared with the performance of these two prediction models.

The following steps are taken:

1. Extract JPEG quantization tables from images
2. Generate feature set for JPEG quantization tables
3. Train decision tree classifier
4. Evaluate classifications
5. Compare with method using hash database

Steps 3 to 5 are performed for the prediction of the camera make and repeated for the prediction of the camera model.

3.2.1 *Extraction of JPEG quantization tables*

As described in Section 2.1, the JPEG quantization tables are used during JPEG compression and relate to the compression ratio of an image. These tables are saved in JFIF headers and can be extracted from the JPEG file. In this research the *djpeg* [1] tool is used. This tool receives an image as input and can output the JPEG quantization tables. These tables are then collected with the use of a python script. The camera make and models are stated in the file names of the images. They are retrieved and are stored together with their JPEG quantization tables for further processing.

3.2.2 *Feature selection*

The decision tree learning algorithm needs attributes of a target as input. The JPEG quantization tables are converted to a feature set which contain all its variables. For example, in the feature set the attribute 'row 1, column 1, luminance' has the value of the variable at the conjunction of the first row and the first column from the luminance quantization table.

As the variables in JPEG quantization tables are somewhat correlated, the hypothesis is made that statistical features of these tables can have an influence on the prediction model. Therefore, extra statistical attributes are added to the feature set. The following values are calculated for each table, for each row and for each column and then added to the feature set: sum, minimum value, maximum value, mean, median, variance, standard deviation.

The assumption is made that not all attributes are evenly important and that the attribute set can contain redundant or irrelevant data. Therefore, feature selection is performed. This selects a subset of relevant features. A tree-based estimator is used to compute feature importances which in turn discards irrelevant features. For this selection, the python Scikit Learn [11] module for tree based feature selection is used.

The decision tree learning algorithm is performed with feature selections on two feature sets: on the set of features that only contains the original attributes from the JPEG quantization table and on the set of features that also contains the extra statistical attributes. This is done in order to analyse if the extra statistical attributes help to create a more accurate decision tree.

3.2.3 *Decision tree learning*

In this research the decision tree learning algorithm is used to create a predictive model. The method of decision tree learning is explained in Section 2.2. The decision tree learning algorithm is a supervised learning algorithm and consist of two stages: training and validation. The dataset is split into a training set and a validation set, which contain the feature sets and their corresponding labels. During the training stage, the decision tree learning algorithm is given the complete training set. It then creates a decision tree based on this set. In the validation stage, this decision tree is evaluated with a validation set. The decision tree receives the feature sets as input and predicts the

[1] http://linux.about.com/library/cmd/blcmdl1_djpeg.htm

corresponding labels. These predications are then compared with the actual labels found in the validation set.

For the implementation of the decision tree learning algorithm the python Scikit Learn [11] module for decision tree classifiers is used. This implementation uses a CART[2] decision tree learning algorithm, which can produce either classification or regression trees. Because the prediction labels are categorical, this algorithm will produce a classification tree.

3.2.4 *Evaluation*

The prediction models are given a total score with the use of the F$_\beta$-score and a stratified k-fold cross-validation. These methods are described below.

3.2.4.1 *F_β-score*

The performance of the prediction models is evaluated with the use of the the F$_\beta$-score. This score is a measure for the accuracy of a test and considers both precision and recall. The β parameter can be set to let the user give more weight to recall ($\beta > 1$) or precision ($\beta < 1$). The formula is described in Equation 3.

$$precision = \frac{|\{\text{ relevant documents}\} \cap \{\text{ retrieved documents}\}|}{|\{\text{ retrieved documents}\}|} \tag{1}$$

$$recall = \frac{|\{\text{ relevant documents}\} \cap \{\text{ retrieved documents}\}|}{|\{\text{ relevant documents}\}|} \tag{2}$$

$$F_\beta = 1 + \beta^2 * \frac{precision * recall}{(\beta^2 * precision) + recall} \tag{3}$$

In this research both precision and recall are important: precision is important to generate a smaller search space, this measure concerns the fraction of retrieved images that are actually correct; recall is important to retrieve all possible incriminating images, this measure concerns the fraction of relevant images that are actually retrieved. With regard to forensic investigations, the recall of images is very important because you want to gather as much incriminating images as possible. For this reason β is set to 2 to give a higher weight to recall. For this reason the F$_\beta$-score is also mentioned as the F$_2$-score in this research.

The F$_2$-score is calculated for every camera make and model and a weighted average of the F$_2$-scores is calculated to evaluate the prediction model. The weighted average for the F$_2$-score of the model is calculated by giving a weight to every F$_2$-score of the classes, which corresponds to the number of instances for each label. This methods holds label imbalance into account. For example, a recall of 90% for label X is more impressive when there are 10,000 images made with camera make/model X in the dataset than when there are 10 relevant images.

3.2.4.2 *Stratified k-fold Cross-Validation*

The prediction models are evaluated with the k-fold cross-validation method. This technique assesses how the results of a statistical analysis will generalize to an independent data set.

This method splits the dataset into k randomly selected subsets, where $k-1$ subsets are used as the training set and 1 subset is used as the validation set. Because there is a label imbalance in the dataset (i.e. the total images from each camera make/model differ widely) a stratified k-fold cross-validation is performed. The stratification makes sure each subset contains the same percentage of samples of each label as the complete set. The prediction models are trained on the training data and evaluated with the test data. This process is repeated k folds, with each of the k subsets used exactly once as the validation set. For every fold, the weighted F_2-score (as explained in Section 3.2.4.1) is calculated. As a final score for the prediction model, the average of these weighted F_2-scores is given.

In this research the value 5 is chosen for k as this splits the dataset such that 80% is used for the training set and 20% is used for the validation set.

3.2.5 *Comparison with hash database*

A simple way of predicting camera make and model according to JPEG quantization tables is to build a database which contains encountered JPEG quantization tables and their corresponding make and model, and then query for the found JPEG quantization tables. Since these tables are comprised of many variables, it is more efficient to store them as a single hash signature. The JPEGSnoop software, for example, works with a database of signatures. In order to evaluate the decision tree learning algorithm, its performance is compared to the performance when a database containing hashes is used. The hash database method is trained and evaluated with the same subsets that are used for the decision tree learning algorithm. If the hash database does not recognize the set of JPEG quantization tables in the evaluation stage, it will randomly chose one of the class labels as its prediction.

The hash database is created by hashing every set of JPEG quantization tables with the SHA256 hashing algorithm and then saving this in the database with its corresponding label. Two different implementations are made:

1. **1→1 Hash Database**: a 1→1 mapping of a set of JPEG quantization tables to 1 camera make/model. The JPEG quantization table is mapped to the first camera make/model that is encountered.

2. **1→ N Hash Database**: a 1→ N mapping of a set of JPEG quantization tables to multiple possible camera make/model. The JPEG quantization table can belong to different camera make/models. This method is also used in the JEGSnoop software.

Both hash database methods are evaluated with the F_2-score as described in Section 3.2.4.

4 RESULTS

In this chapter the results of this research are described and a discussion on these results is given.

4.1 EXTRACTION OF JPEG QUANTIZATION TABLES

In total, there are 1,016 unique sets of JPEG quantization tables retrieved from the images. In Table 4, the number of unique JPEG quantization tables per camera model are shown. As can be derived from these numbers, there are distinct JPEG quantization tables that have been found for multiple camera models. There are 398 quantization tables found in images with different camera models.

As the chrominance color space can be divided into chrominance-red and chrominance-blue, it could have occurred that 3 JPEG quantization tables are retrieved for an image. However, only sets of 2 JPEG quantization tables are found in the dataset.

4.2 FEATURE SELECTION

The two JPEG quantization tables are converted to a set of features and the extra statistical features (as described in Section 3.2.2) are added. Each JPEG quantization table contains 64 values to which 105 extra statistical features per table are added, which gives a total of 338 attributes per image. The extra features had no impact on the scores for the evaluation. Therefore, these extra statistical features are omitted from the feature set.

After the feature selection procedure, the identifiable parameters for the decision tree learning algorithm are reduced to 50 parameters. In Table 5 the importance of every attribute is depicted. The parameters do not show a clear correlation with the tables. The total importance of the luminance JPEG quantization table is 0.56 and total importance of the chrominance JPEG quantization table is 0.44. This shows that both JPEG quantization tables have a comparable importance in this camera identification method.

4.3 DECISION TREE LEARNING

The decision tree learning algorithm has created a decision tree of 603 nodes with a depth of 26 nodes. The average F_2-score is 89% for the prediction of the camera make and 80% for the prediction of the camera model. The exact F_2-scores for every camera make and model are described in Table 6 and Table 7, respectively.

The decision tree model gains high scores for the prediction of the camera make. The mean of F_2-scores for the prediction of the camera make is 85.05% with a standard deviation of 16.32%. There are a few camera makes that have a significantly lower F_2-score, such as Motorola with an F_2-score of 43.30%. This result is unexpected, because the dataset contains 4060 images made by Motorola cameras in which only 6 unique sets of JPEG quantization tables are found. The decision tree only needs to correctly classify a small number of sets (i.e. these 6 unique sets) in its tree for a high F_2-score. This result can be explained by the occurrence of these sets in images from other camera makes. If the set of tables is found more frequently for other camera makes, the decision tree will set the predicted label for these tables for the other camera make. In contrast, the dataset contains 39 unique sets of JPEG quantization tables for 1318 images made by a Panasonic device, which is significantly higher, and this class has an F_2-score of 100%.

The decision tree model also gains a satisfactory F_2-score for the prediction of the camera model. However, at closer inspection the F_2-scores for the classes show a big variance between the classes: the mean of the F_2-scores is 59.30% with a standard deviation of 37.46%. For example, the prediction for camera model Samsung NX1000 gains a F_2-score of 0%. No image that is made with this camera model is correctly classified. There were 4 unique sets of JPEG quantization tables encountered for this camera model in 350 images. This result can also be explained by the occurrence of these tables in images from other camera models of which more examples were present in the dataset. In contrast, the Agfa Sensor505-x camera model gains a F_2-score of 96.29% where the dataset contains 143 unique sets of JPEG quantization tables in 209 images from this camera model.

Because the decision tree model makes a unique map of a set of JPEG quantization tables to a single camera make or model, a trade-off can be found between the F_2-scores. When a set of JPEG quantization tables is encountered for two (or more) different classes, it maps the set to only one of these two (or more) classes. As a result the F_2-score will increase for the class that is mapped to this set and the F_2-score will decrease for the classes that are not mapped to this set. In the prediction of the camera make, the trade-off between different classes is not strongly visible. The conclusion can be drawn that the occurrence of the same set of JPEG quantization tables for different camera models is more often found in images with the same camera make.

Another conclusion that can be drawn is that the presence of sets of JPEG quantization tables in images from other camera makes and models will affect the performance of the decision tree. The number of unique sets of JPEG quantization tables found for a camera make or model does not directly relate to the F_2-score for the prediction model, however, it can increase the probability for a set to be found in images from different camera makes or models and consequently create a trade-off between classes.

4.4 COMPARISON AGAINST HASH DATABASE

The decision tree model is compared to the two hash database models that are explained in Section 3.2.5. They are compared for the prediction of the camera make as well as the prediction of the camera model. An overview of the scores is given in Table 1 and Table 2.

The decision tree model has the highest F_2-score for the prediction of the camera make. For the prediction of the camera model, it scores 3% lower on the F_2-score than the $1 \rightarrow N$ hash database model. Overall, the decision tree model scores better than the $1 \rightarrow 1$ hash database model and comparable to the $1 \rightarrow N$ hash database model.

The 1→ N hash database model scores high on recall for both predictions. This result is explained by the fact that the hash database model stores all possible camera make and models for each unique set of JPEG quantization tables. It returns all possibilities and will receive a high recall as a result of the probability that the correct camera make/model in the returned set is very high. However, it receives low precision rates for the predictions of the camera make as well for the prediction of the camera model. This is also a result of the method returning all possibilities as many false positives are returned.

The 1→1 hash database model has the lowest F_2-scores. This is a result of overfitting of data. This method only returns the first camera make/model class where this set of JPEG quantization tables is seen. The tables are stored as a single hash and consequently it will not recognize a slightly modified set of JPEG quantization tables since this results in a completely different hash.

Algorithm	**Precision**	**Recall**	**F_2-score**
Hash (1→1)	79 %	68 %	68 %
Hash (1→ N)	50 %	99 %	83 %
Decision tree	90 %	89 %	89 %

Table 1: Camera Make Identification

Algorithm	**Precision**	**Recall**	**F_2-score**
Hash (1→1)	54 %	39 %	37 %
Hash (1→ N)	50 %	98 %	83 %
Decision tree	78 %	82 %	80 %

Table 2: Camera Model Identification

4.5 DISCUSSION

Since a decision tree uses a one-to-one mapping from JPEG quantization tables to a camera make/model, it will not perform perfectly on tables that occur at multiple camera make/models. The classifier makes a choice to which camera make/model this table is mapped. In contrast, the 1→ N hash database prediction model returns all possible camera makes/models. This method gains a high recall, but receives a low precision rate. Because all possibilities are returned, instead of only one camera make/model, the search space for these classes is significantly bigger than for decision tree learning. In order to decrease the search space for further processing with other camera identification methods, decision tree learning is preferred because it receives high scores for recall as well as precision.

Although the decision tree learning algorithm is prone to overfitting data, it will more accurately predict camera make and models for sets of JPEG quantization tables that differ slightly. The hash database models will not recognize the set and will return a random value in this implementation. The decision tree model, however, only uses a subset of the features found in the JPEG quantization tables and as a result can handle small differentiations better.

With respect to the creation of the prediction model, the hash database models are more easily trained. When a new set of JPEG quantization tables for a camera make/model occurs, it can be hashed and immediately stored in the database. In contrast, the decision tree model needs to be re-evaluated at the occurrence of a new set because this can result in a different decision tree.

It should be taken into account that the dataset only consists of original images. Image editing software such as Photoshop also use JPEG compression and will contain JPEG quantization tables correlated to Photoshop instead of their original camera make or model. For the decision tree learning algorithm, images that are edited with Photoshop will be predicted as belonging to the same source even though the original images are made with different camera makes and models.

5 CONCLUSION

In this research, a decision tree learning approach is used for camera identification with the use of JPEG quantization tables.

Images are retrieved in forensic investigation as important evidence. In these cases, the origin of these images need to be identified for which camera identification models can be used. The set of JPEG quantization tables utilized for JPEG compression in the camera can be used for these models. The matching process for these tables with an existing database can be time costly when large sets of images are recovered for forensic investigation. For this reason, this research has focused on reducing the search space for JPEG quantization tables. As the camera identification problem can be seen as a pattern recognition problem, machine learning techniques can be applied. The research question on which this research has focused was stated as: *'Can searching through JPEG quantization tables be optimized with the use of decision tree learning?'*

In this research, a decision tree learning approach is taken. Supervised machine learning is used to create a decision tree model to predict camera makes and models on basis of the set of JPEG quantization tables that belong to an image. First, a feature selection procedure is performed in which the identifiable parameters in a set of JPEG quantization tables are reduced from 128 to 50 attributes. These 50 parameters do not show a clear special correlation with the tables.

Then, the decision tree model is trained for the prediction of camera makes as well as camera models and is evaluated using the weighted F_2-score in a 5-fold stratified cross-validation. The decision tree model has a comparable performance in F_2-scores for the predictions of different camera make classes. In contrast, in the prediction for the camera models, the decision tree model performance has a significantly large variance between classes for the F_2-scores. This is a result of the trade-off in performance that takes place between classes. This trade-off occurs less in the prediction of the camera make and it can therefore be concluded the occurrence of the same sets of JPEG quantization tables is more frequent for images from other camera models with the same camera make.

The decision tree model is compared to two different prediction models that use a hash database. The decision tree model gains the highest F_2-score (89%) for the prediction of the camera make. The $1 \rightarrow N$ hash database model performs slightly better for the prediction of the camera make than the decision tree model, with a F_2-score of 83% and F_2-score of 80%, respectively. However, it gains low precision rates because it returns all possible classes. The low precision indicates that the search space is not effectively decreased. In contrast, the decision tree model gains high rates for accuracy while maintaining a high precision. The decision tree model can also handle small differentiations in tables better than the hash database models.

Overall, the decision tree learning algorithm gains a good performance for the prediction of the camera make as well as for the prediction of the camera model and can be used to optimize searching through JPEG quantization tables for camera identification.

5.1 FUTURE WORK

The following adjustments for the methods used in this research, as well as improvements using other techniques, are proposed:

- **Extend image database**: This research can be extended by using a larger image database which comprises of more different camera make/models. It can also be extended with images that are edited with image editing software.

- **Extend feature set**: The feature set can be extended with more attributes that correspond to the image. These can be related to the JPEG quantization tables, but also to other meta-data retrieved from the image.

- **Compare to other learning algorithms**: The decision tree learning algorithm is prone to overfitting. Other supervised machine learning algorithms, such as Naive Bayes and Support Vector Machine, tend to create better generalizations and are less susceptible for overfitting. Future research can be performed on the performance of other machine learning algorithms in comparison with the decision tree learning algorithm.

- **Probabilistic classification**: Because a set of JPEG quantization tables can correspond to multiple camera make/models, this research can be improved with the implementation of probabilistic classification. This method gives a probability distribution over a set of classes instead of predicting one single class for a feature set.

A

APPENDIX A

This appendix contains information on the image database, such as the number of images per camera model and the number of JPEG quantization tables found.

A.1 CAMERA MAKE AND MODEL OF IMAGE DATABASE

Make	**Model**	**# Pictures**
Agfa	DC-504	262
Agfa	DC-733s	329
Agfa	DC-830i	414
Agfa	Sensor505-x	209
Agfa	Sensor530s	406
Blackberry	Curve-9300	1080
Blackberry	Curve-9360	2669
Canon	Ixus55	242
Canon	Ixus70	585
Canon	Powershot-A430	10326
Canon	Powershot-A630	1458
Canon	PowerShot-A640	188
Casio	EX-Z150	946
Casio	EXILIM-EX-FC100	15
Epson	StylusSX205	31
FujiFilm	FinePixJ50	647
Kodak	M1063	2458
Logitech	QuickCam-Communicate-STX	4059
Motorola	V360	4060
Nikon	CoolPixS710	993
Nikon	D200	831
Nikon	D70	405
Nikon	D70s	409
Olympus	mju	1052
Panasonic	DMC-FZ50	962
Panasonic	Lumix-FZ45	356
Pentax	OptioA40	715
Pentax	OptioW60	239
Praktica	DCZ5.9	1039
PS	Vita	220
Ricoh	GX100	1283
Rollei	RCP-7325XS	607
Samsung	Digimax-S500	1060
Samsung	Galaxy-S3-mini	1280
Samsung	L74wide	705
Samsung	NV15	663
Samsung	NX1000	350
Samsung	ST30	340
Sony	DSC-H50	593
Sony	DSC-T77	758
Sony	DSC-W170	422

Table 3: Camera make and model of image database

A.2 NUMBER OF UNIQUE QUANTIZATION TABLES PER CAMERA MODEL

Make	Model	# Unique Tables
Agfa	DC-504	3
Agfa	DC-733s	329
Agfa	DC-830i	87
Agfa	Sensor505-x	143
Agfa	Sensor530s	3
Blackberry	Curve-9300	1
Blackberry	Curve-9360	2
Canon	Ixus55	3
Canon	Ixus70	3
Canon	Powershot-A430	75
Canon	Powershot-A630	1
Canon	PowerShot-A640	1
Casio	EX-Z150	10
Casio	EXILIM-EX-FC100	1
Epson	StylusSX205	1
FujiFilm	FinePixJ50	4
Kodak	M1063	53
Logitech	QuickCam-Communicate-STX	1
Motorola	V360	6
Nikon	CoolPixS710	247
Nikon	D200	45
Nikon	D70	97
Nikon	D70s	102
Olympus	mju	14
Panasonic	DMC-FZ50	37
Panasonic	Lumix-FZ45	2
Pentax	OptioA40	3
Pentax	OptioW60	24
Praktica	DCZ5.9	210
PS	Vita	1
Ricoh	GX100	156
Rollei	RCP-7325XS	3
Samsung	Digimax-S500	22
Samsung	Galaxy-S3-mini	1
Samsung	L74wide	146
Samsung	NV15	18
Samsung	NX1000	4
Samsung	ST30	2
Sony	DSC-H50	11
Sony	DSC-T77	19
Sony	DSC-W170	10

Table 4: Number of Unique Quantization Tables per Camera Model

B

APPENDIX RESULTS

This appendix contains results for the methods used in this research.

B.1 IMPORTANT FEATURES

Luminance							
0.92	0.32	0.33	0.19	0.31	0.12	0.87	0.78
0.40	2.08	1.67	0.16	0.18	0.74	0.21	0.63
0.14	0.12	0.05	0.33	0.32	1.22	0.79	0.41
0.61	0.53	2.83	1.24	0.17	0.32	0.67	1.02
0.90	1.37	2.38	1.33	0.64	1.57	1.45	1.28
0.18	0.48	0.28	1.78	1.63	0.94	1.83	1.71
0.13	0.16	1.32	1.02	0.21	1.15	0.78	1.62
3.09	1.19	0.16	1.41	0.42	1.64	0.16	1.03

Chrominance							
0.90	0.25	0.27	1.68	0.61	1.09	0.28	2.11
0.25	0.76	0.19	1.04	0.83	1.45	0.21	0.14
0.80	0.95	0.36	0.40	0.07	0.05	0.53	1.30
0.49	0.79	0.11	0.02	0.06	1.44	0.32	0.26
0.30	0.26	0.18	0.03	0.93	0.90	0.39	0.07
0.40	0.11	0.52	3.11	0.26	0.79	0.27	1.16
0.14	1.50	0.49	0.51	0.50	1.89	1.54	1.48
1.58	0.21	0.00	0.41	1.09	0.53	0.91	1.56

Table 5: Importance of JPEG quantization tables parameters (multiplied by 100)

B.2 CAMERA MAKE F$_\beta$-SCORES

Camera Make	Fold #1	Fold #2	Fold #3	Fold #4	Fold #5	Average
Agfa	99.95	99.92	99.92	100.00	99.95	99.95
Blackberry	79.87	79.62	81.50	80.77	81.02	80.56
Canon	93.01	93.75	91.76	91.86	93.25	92.72
Casio	84.18	81.83	84.01	82.04	85.03	83.42
Epson	58.76	59.09	61.02	56.16	57.59	58.52
FujiFilm	99.16	98.54	98.33	98.55	98.75	98.67
Kodak	100.00	100.00	100.00	100.00	100.00	100.00
Logitech	93.98	93.20	93.84	93.06	95.73	93.96
Motorola	41.54	44.05	42.20	41.67	47.07	43.30
Nikon	99.72	99.43	99.80	99.31	99.14	99.48
Olympus	83.56	83.80	81.25	83.91	84.14	83.33
PS	100.00	90.91	97.22	96.77	100.00	96.98
Panasonic	100.00	100.00	100.00	100.00	100.00	100.00
Pentax	76.16	75.37	76.43	76.68	78.75	76.68
Praktica	62.06	62.50	65.37	65.27	63.76	63.79
Ricoh	79.30	78.45	79.21	80.68	79.87	79.50
Rollei	99.93	99.93	99.98	99.93	99.98	99.95
Samsung	97.60	97.78	97.99	97.89	97.71	97.79
Sony	68.97	66.09	67.00	68.17	66.82	67.41

Table 6: F$_\beta$-scores in percentages for each camera make for every fold

B.3 CAMERA MODEL F_β-SCORES

Camera Make	**Camera Model**	**Fold #1**	**Fold #2**	**Fold #3**	**Fold #4**	**Fold #5**	**Average**
Agfa	DC-504	34.81	43.48	50.30	52.15	51.08	46.36
Agfa	DC-733s	55.91	52.88	47.08	56.60	47.85	52.07
Agfa	DC-830i	63.05	62.87	64.80	57.44	63.99	62.43
Agfa	Sensor505-x	96.27	96.17	97.38	95.45	96.17	96.29
Agfa	Sensor530s	79.13	75.69	81.24	73.43	75.06	76.91
Blackberry	Curve-9300	61.15	65.63	68.56	62.20	69.93	65.49
Blackberry	Curve-9360	100.00	100.00	100.00	100.00	100.00	100.00
Canon	Ixus55	0.00	0.00	5.10	0.00	0.00	1.02
Canon	Ixus70	86.57	85.95	86.34	84.46	86.21	85.90
Canon	PowerShotA640	0.00	0.00	0.00	0.00	0.00	0.00
Canon	Powershot-A430	94.29	94.58	94.26	94.47	94.52	94.43
Canon	Powershot-A630	0.00	0.00	0.00	0.00	0.00	0.00
Casio	EX-Z150	100.00	100.00	100.00	100.00	100.00	100.00
Casio	EXILIM-EX-FC100	0.00	0.00	0.00	0.00	0.00	0.00
Epson	StylusSX205	96.77	96.77	100.00	100.00	93.75	97.46
FujiFilm	FinePixJ50	93.66	94.03	92.97	94.34	93.93	93.79
Kodak	M1063	99.47	99.35	99.43	99.47	99.43	99.43
Logitech	QuickCam-Communicate-STX	99.72	99.91	100.00	99.72	99.72	99.82
Motorola	V360	100.00	100.00	100.00	100.00	100.00	100.00
Nikon	CoolPixS710	99.93	99.98	99.93	99.98	99.93	99.95
Nikon	D200	91.02	92.39	91.07	92.39	92.06	91.79
Nikon	D70	58.00	53.86	61.78	41.98	47.79	52.68
Nikon	D70s	36.27	34.88	32.98	48.08	48.19	40.08
Olympus	mju	61.57	64.37	63.11	64.20	65.39	63.73
PS	Vita	100.00	100.00	100.00	100.00	100.00	100.00
Panasonic	DMC-FZ50	54.19	45.00	55.29	46.34	40.00	48.16
Panasonic	Lumix-FZ45	53.38	41.16	0.00	52.83	40.27	37.53
Pentax	OptioA40	100.00	100.00	100.00	100.00	100.00	100.00
Pentax	OptioW60	2.54	5.03	5.15	7.65	0.00	4.07
Praktica	DCZ5.9	33.30	37.26	34.64	35.20	39.72	36.03
Ricoh	GX100	93.70	92.35	91.61	92.00	94.42	92.82
Rollei	RCP-7325XS	83.22	83.33	83.33	83.10	85.69	83.74
Samsung	Digimax-S500	92.41	91.81	91.47	91.87	91.23	91.76
Samsung	Galaxy-S3-mini	98.08	98.46	98.69	98.39	98.69	98.46
Samsung	L74wide	5.82	24.18	7.50	22.94	13.25	14.74
Samsung	NV15	0.00	1.88	1.87	0.00	1.84	1.12
Samsung	NX1000	0.00	0.00	0.00	0.00	0.00	0.00
Samsung	ST30	0.00	0.00	31.25	0.00	27.93	11.84
Sony	DSC-H50	2.08	3.08	0.00	1.05	12.89	3.82
Sony	DSC-T77	51.82	58.14	56.12	54.66	53.36	54.82
Sony	DSC-W170	16.25	19.04	14.35	22.39	0.00	14.41

Table 7: F_β-scores in percentages for each camera model for every fold

BIBLIOGRAPHY

[1] *A Focus on Efficiency*. White Paper, September 2013. https://fbcdn-dragon-a.akamaihd.net/hphotos-ak-ash3/851560_196423357203561_929747697_n.pdf.

[2] L. Breiman, J. Friedman, R. Olshen, C. Stone, D. Steinberg, and P. Colla. Cart: Classification and regression trees. *Wadsworth: Belmont, CA*, 156, 1983.

[3] H. Farid. Digital image ballistics from jpeg quantization. Technical report, Dartmouth College, Department of Computer Science, 2006.

[4] H. Farid. Digital image ballistics from jpeg quantization: A followup study. *Department of Computer Science, Dartmouth College, Tech. Rep. TR2008-638*, 2008.

[5] H. Farid. Exposing digital forgeries from jpeg ghosts. *Information Forensics and Security, IEEE Transactions on*, 4(1):154–160, March 2009.

[6] Futuresource Consulting. *Digital cameras in decline, though interchangeable lens segment sees growth*. Press Release, November 2013. http://www.futuresource-consulting.com/2013-11Cameraspressrelease.html.

[7] Gartner. *Gartner Says Annual Smartphone Sales Surpassed Sales of Feature Phones for the First Time in 2013*. Press Release, February 2013. http://www.gartner.com/newsroom/id/2665715.

[8] T. Gloe and R. Böhme. The 'Dresden Image Database' for benchmarking digital image forensics. In *Proceedings of the 25th Symposium On Applied Computing (ACM SAC 2010)*, volume 2, pages 1585–1591, 2010.

[9] J. D. Kornblum. Using jpeg quantization tables to identify imagery processed by software. *digital investigation*, 5:S21–S25, 2008.

[10] W. Luo, Z. Qu, F. Pan, and J. Huang. A survey of passive technology for digital image forensics. *Frontiers of Computer Science in China*, 1(2):166–179, 2007.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[12] T. Van Lanh, K.-S. Chong, S. Emmanuel, and M. S. Kankanhalli. A survey on digital camera image forensic methods. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 16–19. IEEE, 2007.

[13] G. K. Wallace. The jpeg still picture compression standard. *Consumer Electronics, IEEE Transactions on*, 38(1):xviii–xxxiv, 1992.