Hacettepe University

Computer Engineering Department

BBM465 Information Security Lab. - 2022 Fall

Assignment 4
Anti-Phishing Threat Intelligence

C# / .NET Framework v4.5

January 3, 2023

Students' names:	Students' numbers:
Uğurcan ERDOĞAN	21827373
Burak DAĞLAR	21827302

1. Problem Definition

In this experiment, we are asked to do visual analysis using image files consisting of screenshots. With the threat intelligence module we produced for anti-phishing, we completed website brand classification with screenshots of phishing websites. Similar module ideas are frequently used in real-world cybersecurity applications today.

2. Aim and Solution Implementation

What is Phishing?

Phishing is a form of social engineering where attackers deceive people into revealing sensitive information or installing malware such as ransomware. Phishing attacks have become increasingly sophisticated and often transparently mirror the site being targeted, allowing the attacker to observe everything while the victim is navigating the site, and transverse any additional security boundaries with the victim. As of 2020, it is the most common type of cybercrime, with the FBI's Internet Crime Complaint Centre reporting more incidents of phishing than any other type of computer crime. The term "phishing" was first recorded in 1995 in the cracking toolkit AOHell, but may have been used earlier in the hacker magazine 2600. It is a variation of fishing and refers to the use of lures to "fish" for sensitive information. [1]

We were asked to use C# and .NET framework 4.5 version in our implementation. Based on these requirements, extra NuGet packages were also used in our experiment. The dataset we use is proposed in the study named ”Phish-IRIS: A New Approach for Vision Based Brand Prediction of Phishing Web Pages via Compact Visual Descriptors”. The dataset involves 1313 training and 1539 testing samples distributed among 14 (13 brands such as DHL, Facebook, Yahoo) + 1 legitimate group named ”other”) classes. [2]

In this experiment, the process of identifying screenshots taken from phishing websites with image classification methods is fundamental. The most commonly used method in these processes is to get help from Artificial Neural Networks and their varieties (CNN etc.). However, this is not the only method. Classical **Machine Learning** algorithms can also perform such operations. To benefit from ML algorithms in our experiment, we first need certain tools to make all images suitable for machine learning algorithms, that is, to be able to represent them properly. The tools that provide these appropriate representations are the **Image (Visual) Descriptors** themselves.

In computer vision, visual descriptors or image descriptors are descriptions of the visual features of the contents in images, videos, or algorithms or applications that produce such descriptions. They describe elementary characteristics such as the shape, the color, the texture or the motion, among others. [3]

Visual descriptors make an image or image segment discriminative representable in a vector-based fashion. If the image part mentioned is the whole image, our descriptor will work as a global feature descriptor. However, if the image part consists of certain keypoints in the image, that is, meaningful and distinctive parts, local feature descriptors perform this operation. In our experiment, we used FCTH and CEDD descriptors as global feature descriptor and SURF descriptor as local feature descriptor.

NuGet packages and versions used in the project:

```xml
<packages>
  <package id="Accord" version="3.8.0" targetFramework="net45" />
  <package id="Accord.Imaging" version="3.8.0" targetFramework="net45" />
  <package id="Accord.MachineLearning" version="3.8.0" targetFramework="net45" />
  <package id="Accord.Math" version="3.8.0" targetFramework="net45" />
  <package id="Accord.Statistics" version="3.8.0" targetFramework="net45" />
</packages>
```

Brief Information About Visual Descriptors

FCTH combines the color and texture information, in one histogram. FCTH - Fuzzy Color and Texture Histogram - results from the combination of 3 fuzzy systems. FCTH size is limited to 72 bytes per image, rendering this descriptor suitable for use in large image databases. The proposed feature is appropriate for accurately retrieving images even in distortion cases such as deformations, noise and smoothing. The configuration of FCTH is resolved as follows: The histogram is constituted by 8 regions, as these are determined by the fuzzy system that takes decision with regards to the texture of the image. Each region is constituted by 24 individual regions, as these results from the second fuzzy system. Overall, the output that results includes 8×24=192 bins. Based on the content of the bins the respecting final histogram is produced. [4]

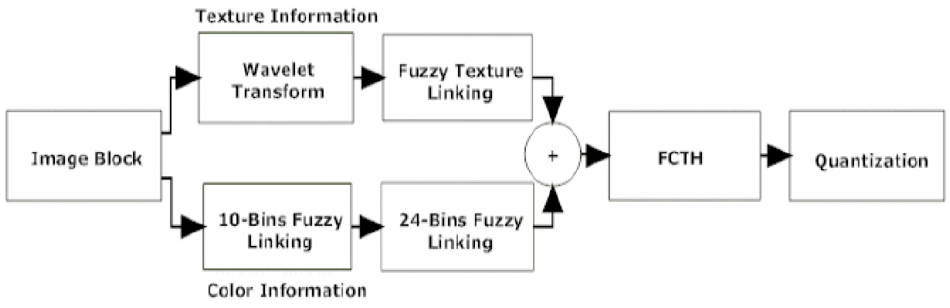


Figure 1: Combination of the three fuzzy systems of FCTH.[4]

CEDD "Color and Edge Directivity Descriptor" can be used for incorporating color and texture information in a histogram. CEDD size is limited to 54 bytes per image, rendering this descriptor suitable for use in large image databases. One of the most important attribute of the CEDD is the low computational power needed for its extraction, in comparison with the needs of the most MPEG-7 descriptors. The unit associated with the extraction of color information is called Color Unit. Similarly, the Texture Unit is the unit associated with the extraction of texture information. The CEDD histogram is constituted by 6 regions, determined by the Texture Unit. Each region is constituted by 24 individual regions, emanating from the Color Unit. Overall, the final histogram includes 6 × 24 = 144 regions. [5]

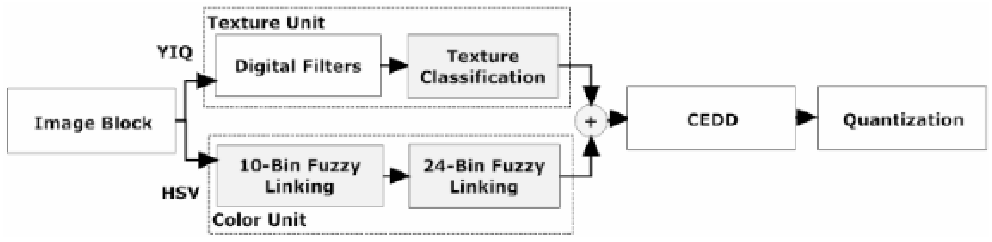


Figure 2: CEDD Flowchart.[5]

SURF is a patented local feature detector and descriptor. It can be used for tasks such as object recognition, image registration, classification, or 3D reconstruction. It is partly inspired by the scale-invariant feature transform (SIFT) descriptor. The standard version of SURF is several times faster than SIFT and claimed by its authors to be more robust against different image transformations than SIFT. To detect interest points, SURF uses an integer approximation of the determinant of Hessian blob detector, which can be computed with 3 integer operations using a precomputed integral image. Its feature descriptor is based on the sum of the Haar wavelet response around the point of interest. These can also be computed with the aid of the integral image. SURF descriptors have been used to locate and recognize objects, people or faces, to reconstruct 3D scenes, to track objects and to extract points of interest. The image is transformed into coordinates, using the multi-resolution pyramid technique, to copy the original image with Pyramidal Gaussian or Laplacian Pyramid shape to obtain an image with the same size but with reduced bandwidth. This achieves a special blurring effect on the original image, called Scale-Space and ensures that the points of interest are scale invariant. [6]

3. Analysis and Results

In the first part of the project, that is, in the "precompute" stage, the output representations that all descriptors will prepare for each image are recorded in the form of a table. At this stage, 3 descriptors will perform operations on the images in the train and val folders, and a total of 6 different CSV files will be produced. In the content of these files, there will be as many columns as the number of output features and additionally the label of the image. In SURF descriptor, however, since it would not be correct to use the outputs directly, we need to save them after certain processes. In this way, each image will be represented more logical for classification tasks.

According to the definition of feature descriptors, we can make the following inferences. The FCTH global descriptor looks at the colors and the texture of the entire and produces a histogram of 192 features. CEDD, on the other hand, produces a histogram of 144 features while addressing the colors and edges of the entire image. SURF, on the other hand, can reveal histograms (vector-based representations) in very different numbers consisting of 64 or 128 features, depending on the variety of key points in the image, unlike global descriptors.

In the project, a 2D array with N x 64 columns is produced from the SURF descriptor per image. Certain operations must be performed on the 2D array space to be produced from all images. Because images with N representations need to be represented in only 1 way (64x1-dim). In addition, since Bag Of Visual Words will be created for each image, a histogram of 400x1 specific to each image must be established. This 400 number is given to us as a constant.
Since 400 centroids will be involved in the KMeans clustering process, we can express the width of the vocabulary with the number 400. The higher this number, the more types of groups there will be and a better grouping will be possible. However, computational costs arise in return.

After the steps below, the final SURF table of 400 features + label for each image will be ready. We can use this table for classification operations.

1. Gathering and accumulating local feature arrays of all images.

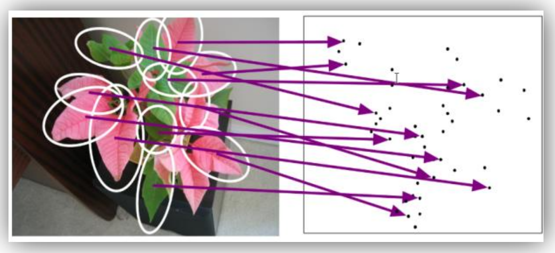
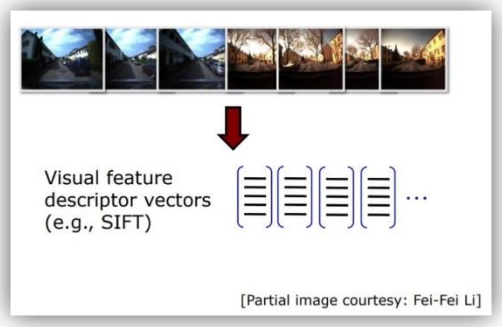


Visual feature descriptor vectors (e.g., SIFT)
[Partial image courtesy: Fei-Fei Li]

Figure 3: SURF feature extracting visualized.[7]

2. Clustering with all of these accumulated arrays. (**QUANTIZATION**)
3. Saving centroids resulting from clustering.

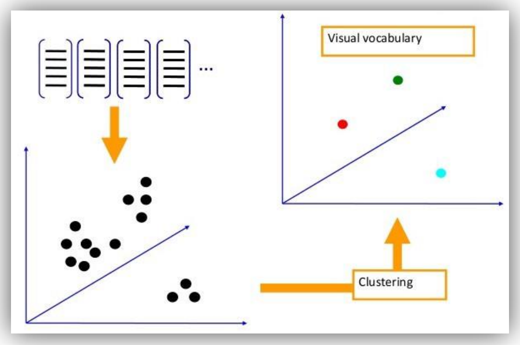


Figure 4: KMeans clustering visualized.[7]

NOTE: We tried working with **MiniBatchKMeans** in Accord.NET.

Instead of working fast as promised, it worked very slowly and strangely, the clustering did not end.

4. Then again, clustering the local feature arrays from each image according to the relevant centroid. (**POOLING**)
5. Saving the resulting cluster group label by increasing the corresponding feature value (the cluster group's slot (centroid value)) in the image's histogram by 1.

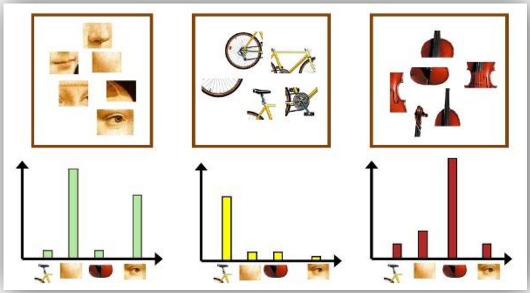
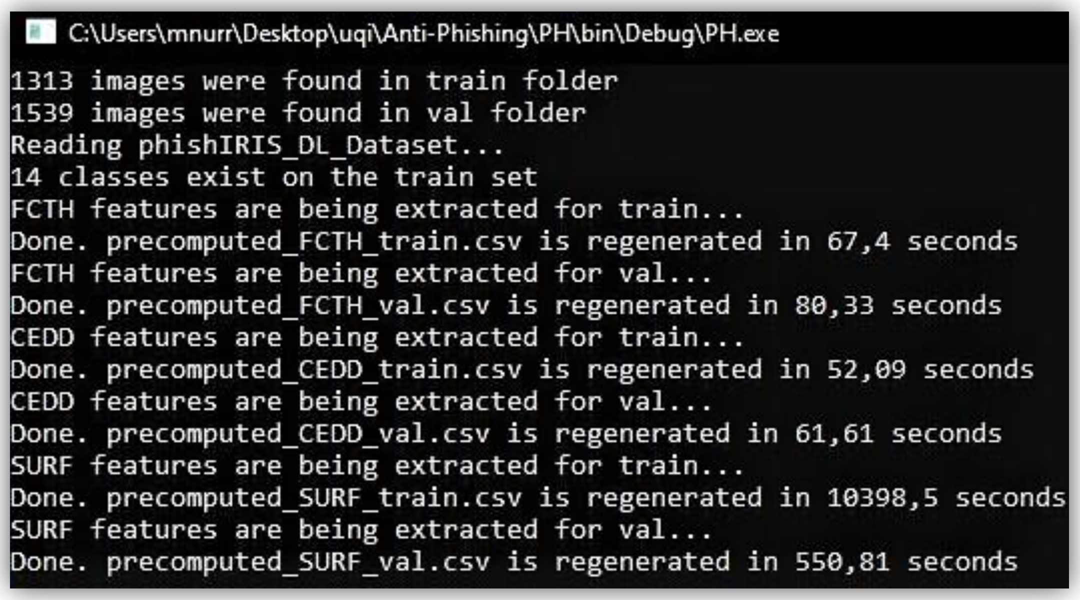


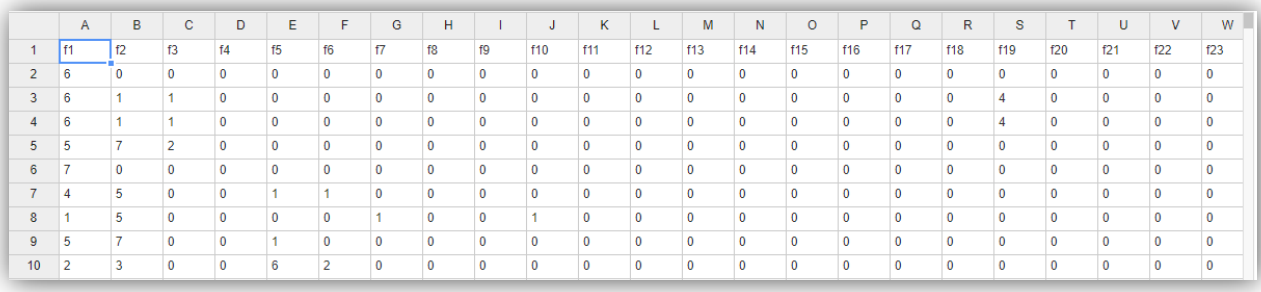
Figure 5: Pooling operation for each image. (Binning).[7]

Finally, we normalized the 2-dimensional Bag Of Visual Words array we created for SURF descriptors, making it more feasible for the other mode of this project, “trainval”, in which classifiers will take place. Outputs of the program and some of the resulting tables are as follows:



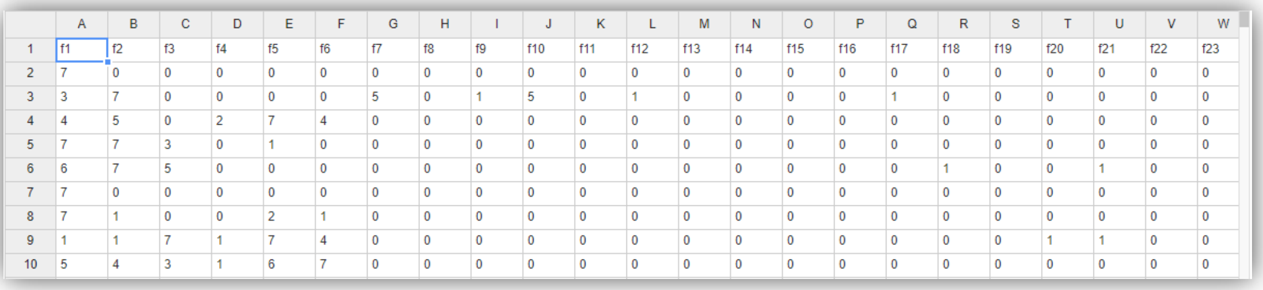
```
C:\Users\mnurr\Desktop\uqi\Anti-Phishing\PH\bin\Debug\PH.exe
1313 images were found in train folder
1539 images were found in val folder
Reading phishIRIS_DL_Dataset...
14 classes exist on the train set
FCTH features are being extracted for train...
Done. precomputed_FCTH_train.csv is regenerated in 67,4 seconds
FCTH features are being extracted for val...
Done. precomputed_FCTH_val.csv is regenerated in 80,33 seconds
CEDD features are being extracted for train...
Done. precomputed_CEDD_train.csv is regenerated in 52,09 seconds
CEDD features are being extracted for val...
Done. precomputed_CEDD_val.csv is regenerated in 61,61 seconds
SURF features are being extracted for train...
Done. precomputed_SURF_train.csv is regenerated in 10398,5 seconds
SURF features are being extracted for val...
Done. precomputed_SURF_val.csv is regenerated in 550,81 seconds
```

Figure 6: Program outputs for the first mode “precompute”.



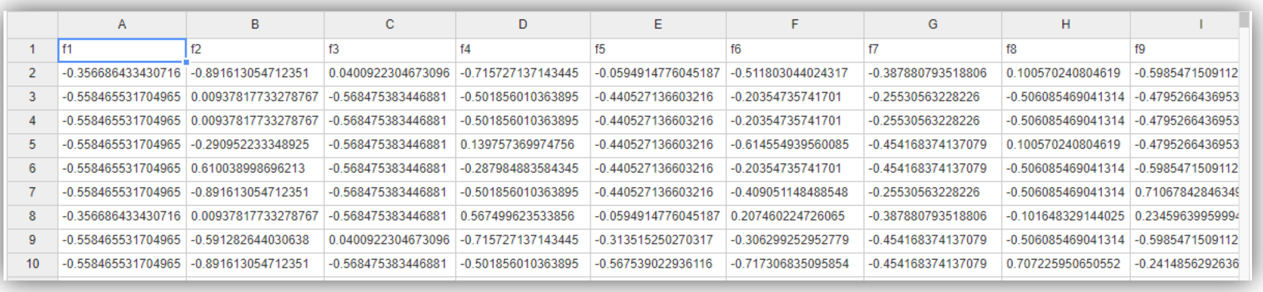
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23
2	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	6	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
4	6	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0
5	5	7	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	4	5	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	5	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	5	7	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	2	3	0	0	6	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7: First 10 row of the precomputed_CEDD_train.csv file.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23
2	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	3	7	0	0	0	0	5	0	1	5	0	1	0	0	0	0	1	0	0	0	0	0	0
4	4	5	0	2	7	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	7	7	3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	6	7	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
7	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	1	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	1	1	7	1	7	4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
10	5	4	3	1	6	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8: First 10 row of the precomputed_FCTH_val.csv file.



	A	B	C	D	E	F	G	H	I
1	f1	f2	f3	f4	f5	f6	f7	f8	f9
2	-0.356686433430716	-0.891613054712351	0.0400922304673096	-0.715727137143445	-0.0594914776045187	-0.511803044024317	-0.387880793518806	0.100570240804619	-0.5985471509112
3	-0.558465531704965	0.00937817733278767	-0.568475383446881	-0.501856010363895	-0.440527136603216	-0.20354735741701	-0.25530563228226	-0.506085469041314	-0.4795266436953
4	-0.558465531704965	0.00937817733278767	-0.568475383446881	-0.501856010363895	-0.440527136603216	-0.20354735741701	-0.25530563228226	-0.506085469041314	-0.4795266436953
5	-0.558465531704965	-0.290952233348925	-0.568475383446881	0.139757369974756	-0.440527136603216	-0.614554939560085	-0.454168374137079	0.100570240804619	-0.4795266436953
6	-0.558465531704965	0.610038998696213	-0.568475383446881	-0.287984883584345	-0.440527136603216	-0.20354735741701	-0.454168374137079	-0.506085469041314	-0.5985471509112
7	-0.558465531704965	-0.891613054712351	-0.568475383446881	-0.501856010363895	-0.440527136603216	-0.409051148488548	-0.25530563228226	-0.506085469041314	0.7106784284634
8	-0.356686433430716	0.00937817733278767	-0.568475383446881	0.567499623533856	-0.0594914776045187	0.207460224726065	-0.387880793518806	-0.101648329144025	0.2345963995999
9	-0.558465531704965	-0.591282644030638	0.0400922304673096	-0.715727137143445	-0.313515250270317	-0.306299252952779	-0.454168374137079	-0.506085469041314	-0.5985471509112
10	-0.558465531704965	-0.891613054712351	-0.568475383446881	-0.501856010363895	-0.567539022936116	-0.717306835095854	-0.454168374137079	0.707225950650552	-0.2414856292636

Figure 9: First 10 row of the precomputed_SURF_train.csv file.

In the <u>second</u> part of the project, that is, in the "trainval" mode, it is necessary to read the previously saved csv files and perform classification operations with these files. Among the classifiers, **Support Vector Machine** and **Random Forest** algorithms are mandatory. The third chosen **C4.5** algorithm is a <u>Decision Tree-based algorithm like Random Forest</u>.

Random Forest and C4.5 algorithms can automatically balance datasets (distribution dependent) when a class is more infrequent than other classes in the data. C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes (**PRUNING**) [8] [9]. But of course, since Random Forest basically uses more than one decision tree and is an Ensemble Learning method, it is expected to give better results than the C4.5 algorithm. The reason for choosing SVM is that this classifier is non-parametric, except the kernel choice, and is more effective in high-dimensional spaces. It is also relatively memory efficient. [10]

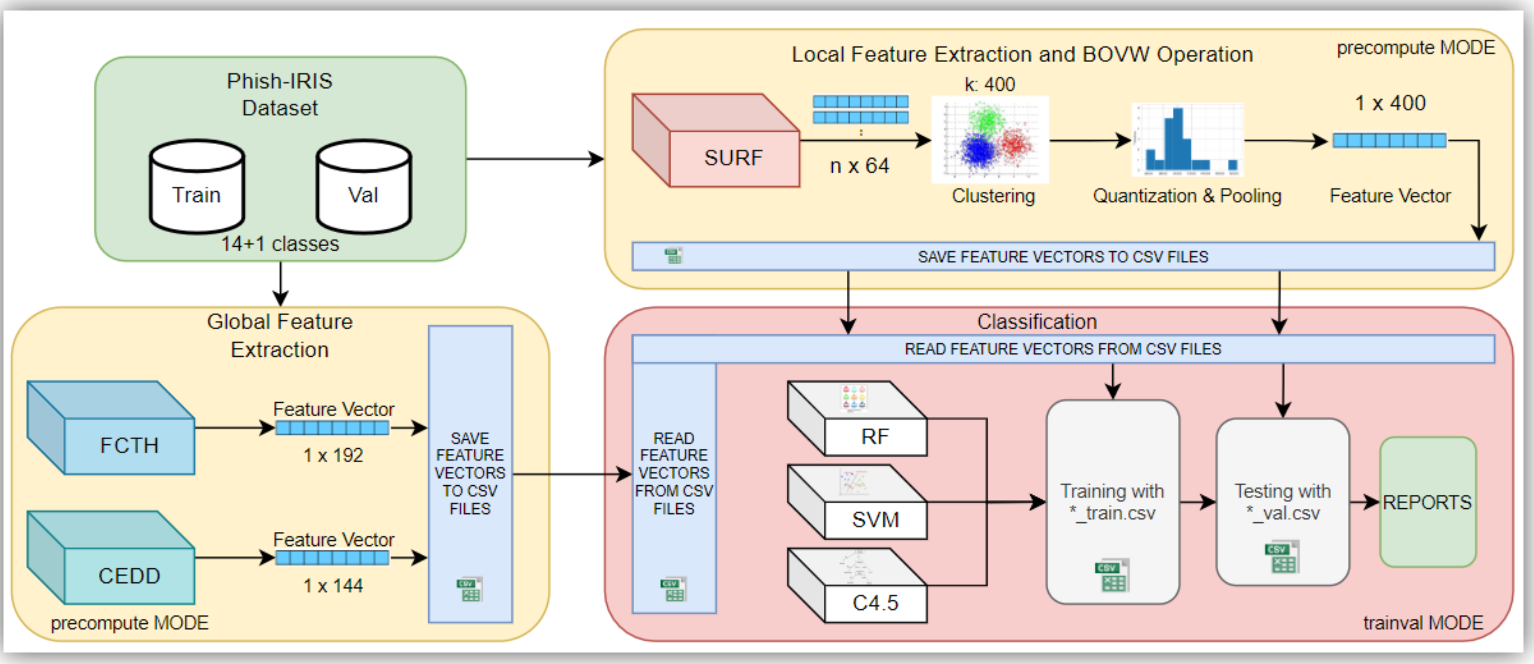


Figure 10: System flowchart with MODES: precompute/trainval

<u>You can find the necessary files/folders in these paths:</u>

ph.exe → ..\PH\PH\bin\Debug\ph.exe

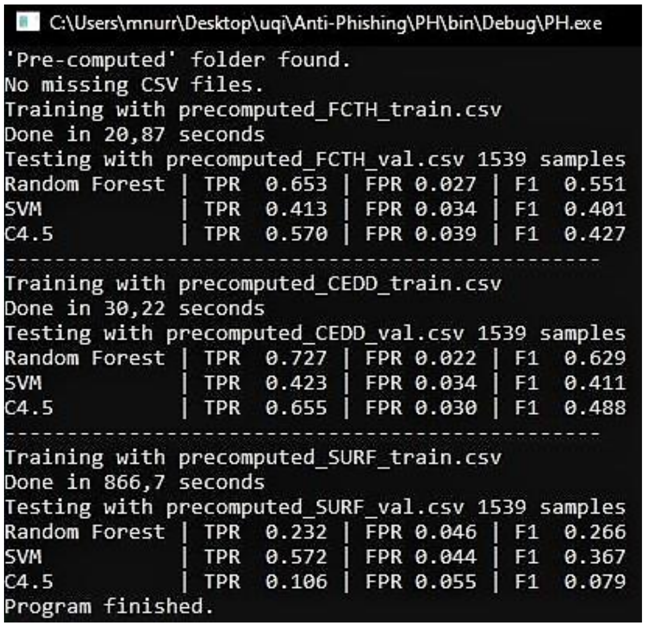
directory of precomputed CSV files → ..\PH\PH\bin\Debug\pre-computed

(dataset → ..\PH\PH\bin\phishIRIS_DL_Dataset*)*

<u>You can run the application "ph.exe" with the following args:</u>

- `ph.exe -dataset ..\phishIRIS_DL_Dataset -mode` **`precompute`**
- `ph.exe -dataset ..\phishIRIS_DL_Dataset -mode` **`trainval`**

We analyzed the performance metrics by giving the train and val sample-label tuples we read from each CSV file to the classifiers. The results of the performance metrics we obtained are as follows:



```
C:\Users\mnurr\Desktop\uqi\Anti-Phishing\PH\bin\Debug\PH.exe
'Pre-computed' folder found.
No missing CSV files.
Training with precomputed_FCTH_train.csv
Done in 20,87 seconds
Testing with precomputed_FCTH_val.csv 1539 samples
Random Forest | TPR  0.653 | FPR 0.027 | F1  0.551
SVM           | TPR  0.413 | FPR 0.034 | F1  0.401
C4.5          | TPR  0.570 | FPR 0.039 | F1  0.427
-----------------------------------------------
Training with precomputed_CEDD_train.csv
Done in 30,22 seconds
Testing with precomputed_CEDD_val.csv 1539 samples
Random Forest | TPR  0.727 | FPR 0.022 | F1  0.629
SVM           | TPR  0.423 | FPR 0.034 | F1  0.411
C4.5          | TPR  0.655 | FPR 0.030 | F1  0.488
-----------------------------------------------
Training with precomputed_SURF_train.csv
Done in 866,7 seconds
Testing with precomputed_SURF_val.csv 1539 samples
Random Forest | TPR  0.232 | FPR 0.046 | F1  0.266
SVM           | TPR  0.572 | FPR 0.044 | F1  0.367
C4.5          | TPR  0.106 | FPR 0.055 | F1  0.079
Program finished.
```

Figure 11: Program outputs and metrics for the second mode

The evaluation has been carried out by considering the metrics of accuracy, true positive rate (TPR), false positive rate (FPR), and F-1 (geometric mean of precision and recall) measure.

- When we look at the results based on the <u>descriptor</u>, we see that we get the best results from CEDD. All of the classifiers, especially Random Forest, give the best metric results in the vectors produced by the CEDD descriptor.
- When we need to interpret it <u>specifically</u>, the C4.5 Decision Tree algorithm gave its best result in CEDD, while it performed very poorly in the output of SURF, which is the local feature descriptor. However, SURF descriptor seems quite suitable for SVM classifier.

- When we look at the <u>time spent in training</u>, it is possible to prefer CEDD. The training process for classifying the SURF vectors took about 15 minutes. In others, this time is on average half a minute.

  It should be noted that global feature descriptors can work very quickly in "precomputed" mode too, and CSV files are produced in about a minute. However, in SURF, CSV files are produced in 2-3 hours. (**Times may vary depending on computer specifications*.)

4. Conclusion and Discussion

We mentioned that we get the best results in the CEDD global descriptor. We think that the main reason for this is the dataset we are working with. Because the logos on the websites in our dataset are specific brand logos and necessary discrimination will be possible when the appropriate descriptor is found. This can lead us to the most accurate result. CEDD and FCTH, on the other hand, were successful because they took Color and Texture details into account. The colors and edge structures of the logos of the websites are different from each other, which has enabled CEDD to be successful. FCTH uses the same fluffy color scheme used by CEDD, but it uses a more detailed edge definition with 8 bins, resulting in a 192-bin overall function. When we investigated the reason why it gave worse results even though it was more detailed, we encountered studies that had similar results. For example, in this comparative study [11], CEDD always gives better results than FCTH. But we can say that, their performance is competitive. In advanced usage, it is possible to combine the forces of FCTH and CEDD thanks to a different global descriptor called Joint Composite Descriptor (JCD). It can also be used.

NOTE: Due to the Accord.NET we used, our Random Forest algorithm was giving an error because some columns in the output arrays from the global descriptors were constant numbers. We continued the process by dropping these columns.

We think that the reason why SURF gives worse results than globals is some errors in our implementation and the lack of necessary key points. In addition, the SURF library we use gives us Nx64-sized outputs instead of giving us Nx128-sized outputs. The decrease in distinctiveness, for these reasons, can also be shown among the things that affect performance. It should also be noted that colors are not involved in SURF. Colors can play an important role in making certain distinctions. Additionally, discriminativeness may increase when we increase the visual word count. Clustering with numbers greater than 400 can be tried to improve performance. As a further suggestion, we can try to classify the BoVW we created after SURF without normalizing it. The metrics that will emerge in this form can also be taken into account. **Our experiment definitely needs improvement**, as we expect local feature descriptors to outperform globals.

REFERENCES

[1] Phishing. (2023, January 1). In *Wikipedia*.
https://en.wikipedia.org/wiki/Phishing

[2] Phish-IRIS dataset
https://web.cs.hacettepe.edu.tr/~selman/phish-iris-dataset/

[3] Visual descriptor. (2022, April 4). In *Wikipedia*.
https://en.wikipedia.org/wiki/Visual_descriptor

[4] S. A. Chatzichristofis and Y. S. Boutalis, "FCTH: Fuzzy Color and Texture Histogram - A Low Level Feature for Accurate Image Retrieval," 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services, 2008, pp. 191-196, doi: 10.1109/WIAMIS.2008.24.
https://ieeexplore.ieee.org/document/4556917

[5] Chatzichristofis, Savvas & Boutalis, Yiannis. (2008). CEDD: Color and Edge Directivity Descriptor: A Compact Descriptor for Image Indexing and Retrieval. 312-322. 10.1007/978-3-540-79547-6_30.
https://www.researchgate.net/publication/221410112_CEDD_Color_and_Edge_Directivity_Descriptor_A_Compact_Descriptor_for_Image_Indexing_and_Retrieval

[6] Speeded up robust features. (2023, January 1). In *Wikipedia*.
https://en.wikipedia.org/wiki/Speeded_up_robust_features

[7] Bag Of Visual Words
https://medium.com/analytics-vidhya/bag-of-visual-words-bag-of-features-9a2f7aec7866

[8] C4.5 algorithm. (2022, February 10). In *Wikipedia*.
https://en.wikipedia.org/wiki/C4.5_algorithm

[9] Understanding C4.5 Decision tree algorithm
https://sumit-kr-sharma.medium.com/understanding-c4-5-decision-tree-algorithm-3bf0981faf4f

[10] Top 4 advantages and disadvantages of Support Vector Machine or SVM
https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107

[11] Alreshidi, E.; Ramadan, R.A.; Sharif, M.H.; Ince, O.F.; Ince, I.F. A Comparative Study of Image Descriptors in Recognizing Human Faces Supported by Distributed Platforms. *Electronics* **2021**, *10*, 915.
https://doi.org/10.3390/electronics10080915