UNKNOWN SHOE SOLE MATCHING

INTRODUCTION, AIMS, OBJECTIVES

What?

- Aim to create a classification system for categorizing unknown shoe sole images.
- construct a pipeline to automatically source data, extract key information, and make predictions
- Experiment with various techniques, including SIFT and deep learning, to determine effective classification approaches, and also provide a benchmark
- Determining if we can enhance recognition accuracy through feature fusion or combining classification approaches

Why?

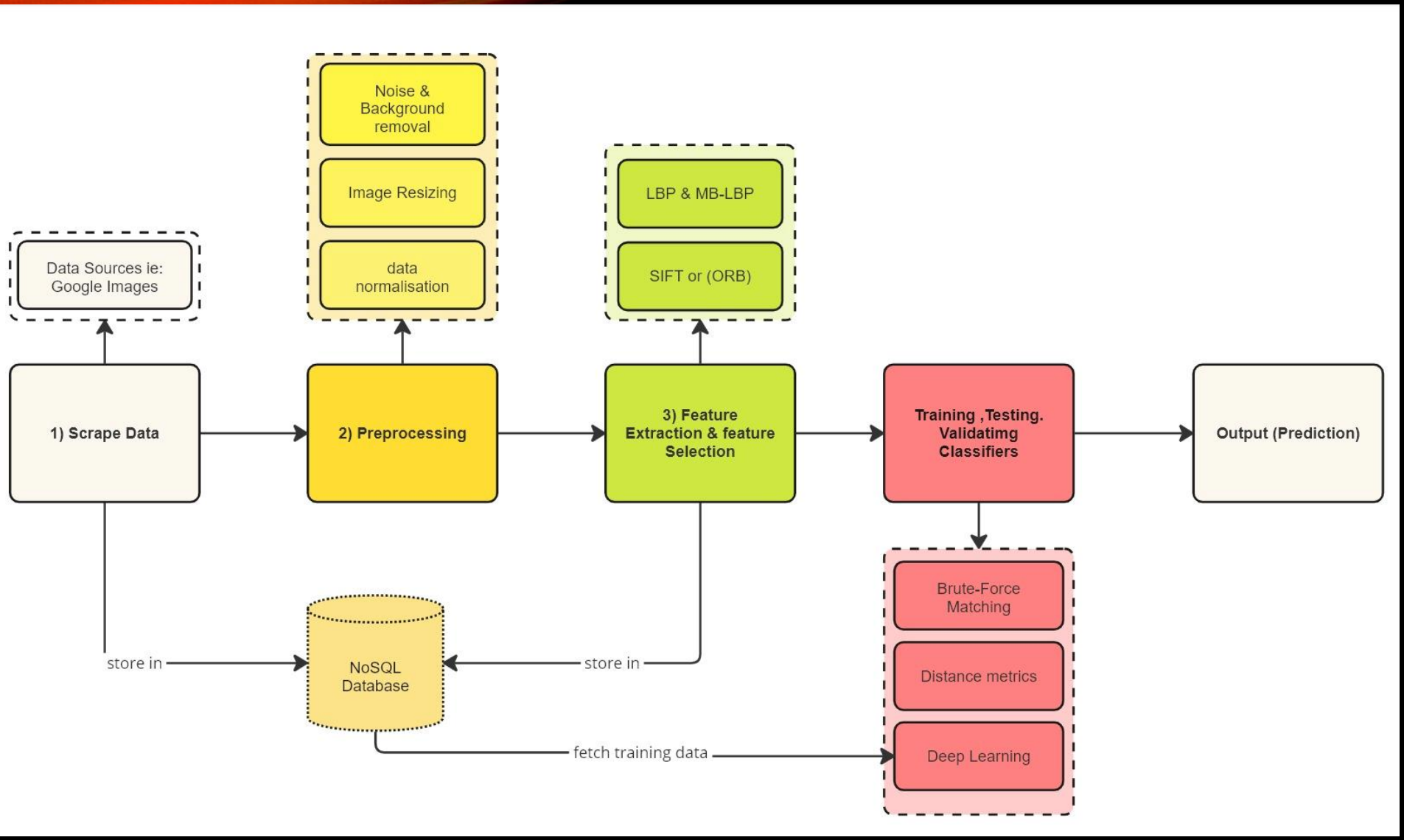
- Essential due to the lack of regulation in new shoe models and patterns, and keeping up with sheer volume of new models
- Current systems updated in periodic cycles (i.e., SoleMate), not keeping pace with shoe releases

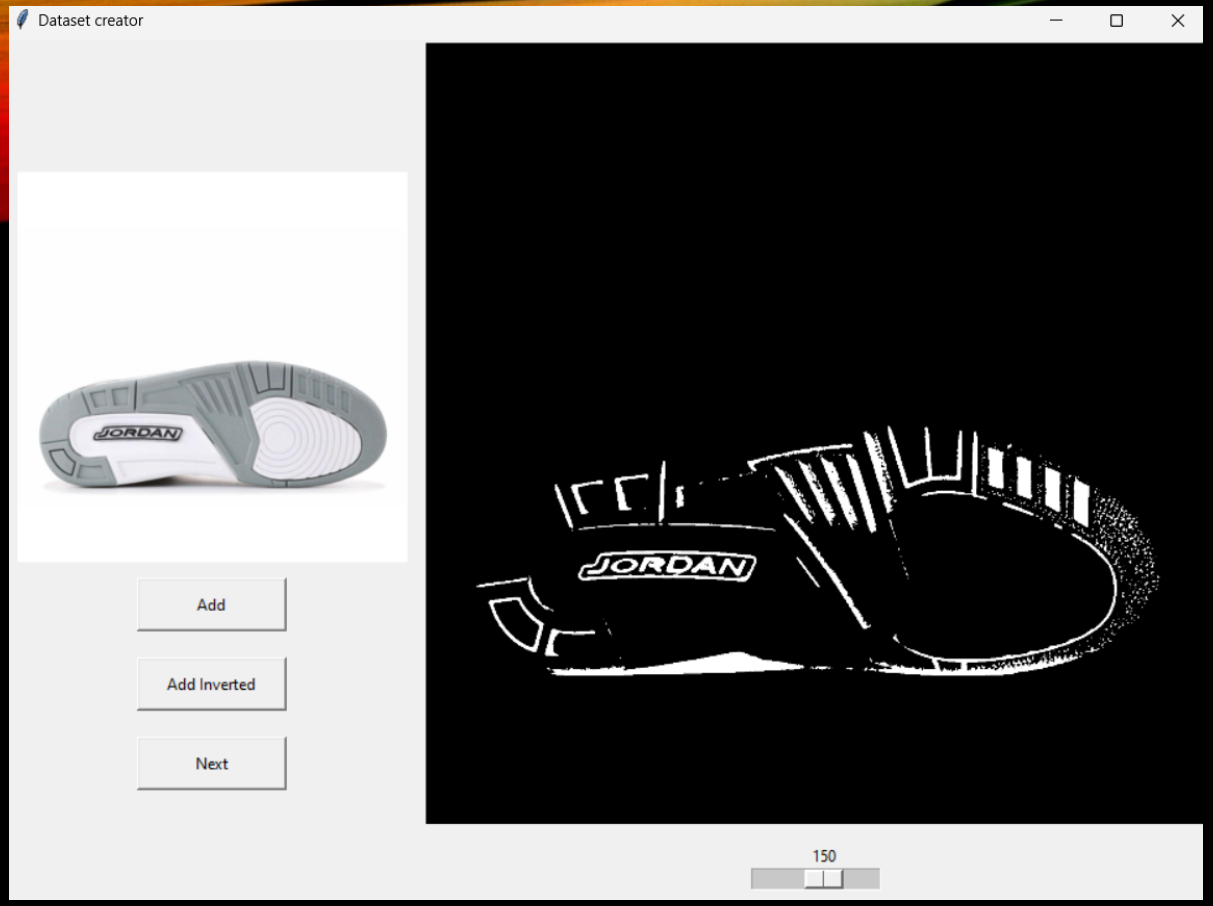
- Automation of shoeprint matching crucial in forensic investigations, providing insights into crime scenes, aiding in identifying shoe make, model, and size.

REQUIREMENTS & CONSTRAINTS

- **To achieve this…**
  - *Dataset requirements: aiming to enforce consistency in the dataset and minimise noise*
  - *Algorithm choice is important: we aim to balance between "exploration" and "exploitation" and provide comparative analysis*
  - *Build scalability in, providing means to incorporate more shoe image data*
- **Constraints**
  - *Limited to sneaker shoes, more specifically Jordan franchise*
  - *Filtering web scraped dataset to be performed manually*
  - *Computational limitations, especially when working with larger CNN models*
  - *Dataset size, containing 1860 samples, though far more data items would be required to improve performance*

SYSTEM OVERVIEW





DATA PREPROCESSING

Operations include:

- Greyscale conversion – simplifying calculations
- Rescaling while maintaining aspect ratios
- Cropping using bounding box, Otsu threshold
- Normalisation
- Aggregate masks
- Data augmentation to bolster dataset size and variation for robustness

FEATURE EXTRACTION

Fig.1, SIFT Key points mapped onto images

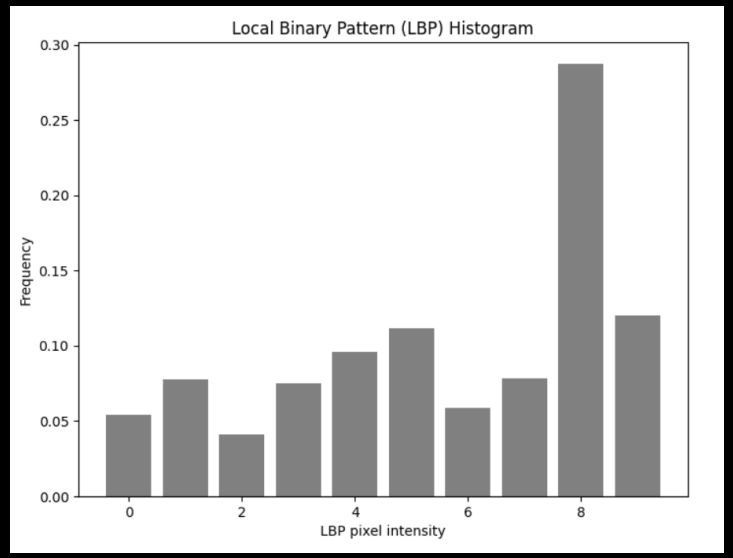


Fig.2, ;bp histogram vectors

How do they work?

- create robust descriptors that are invariant to geometric transformations, scale, and illumination variations.
- Its four-stage process includes space-scale extrema detection, key point localization, orientation assignment, and key point descriptor formation.
- limited to SIFT key point detections to 1000, balancing most insightful information whilst aiming minimising memory use and downstream time costs for matching
- All image augmentations are Brute force matched using threshold value of 0.9, yields optimal accuracy after iterative adjustments to enforce selection criteria.

- LBP : a computationally efficient texture analysis technique, that uses thresholding pixel intensities in a circular region to produce LBP codes that when concatenated form LBP feature histogram
- Various distance measures were tested for histogram comparison: Euclidean, Chi, and Kullback Leibler (KL). KL divergence outperformed others, capturing shape and magnitude differences in distributions.

CNNS

Why?

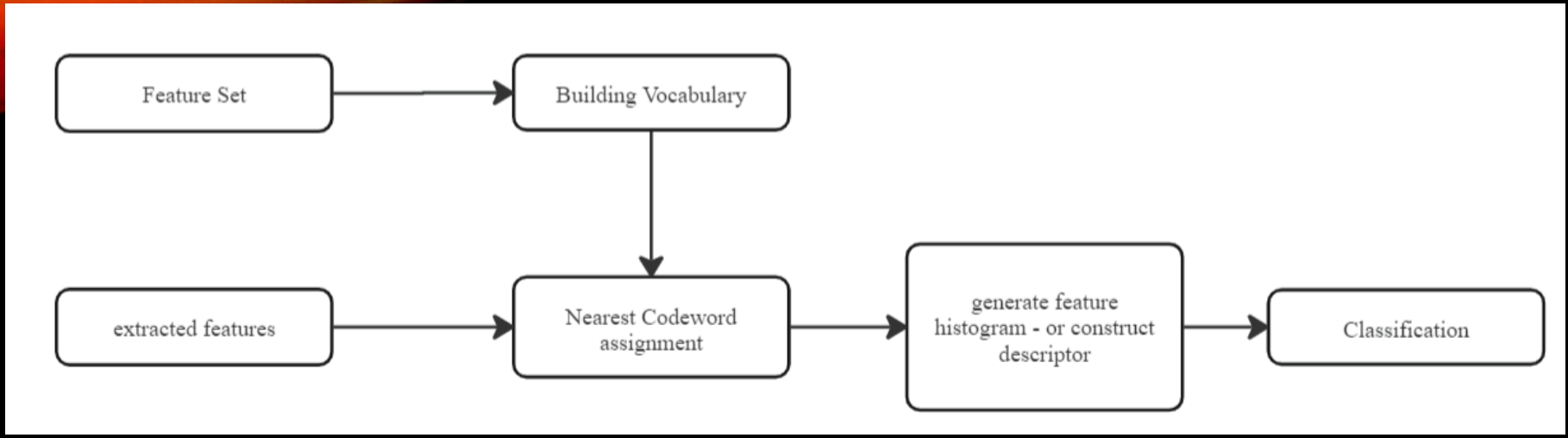
- Can represent Complex functions can through multiple non-linear module combinations to transform representations into higher-level features, capturing hierarchical and abstract features effectively.
- But, we risk overfitting due to limited sized datasets cases of data

How will we be using them?

- Will be used to classify raw image data, SIFT & LBP descriptors,
- Will use different CNNS types, structures and hyperparameters according to dataset characteristics.(i.e., 1D CNNs or 2D CNNs)
- The models we used: SIFT-CNNs, LBP-CNN, CNN(trained on normal images), pretrained CNN
- Hypermeters tuning performed using random grid search

- Pretrained CNNs also used, leveraging prelearned generalisations ,allowing us to focus limited dataset on finetuning the network rather than learnring simple representations

HOW DO WE MAKE OUR DESCRIPTORS COMPATIBLE WITH CNNS?

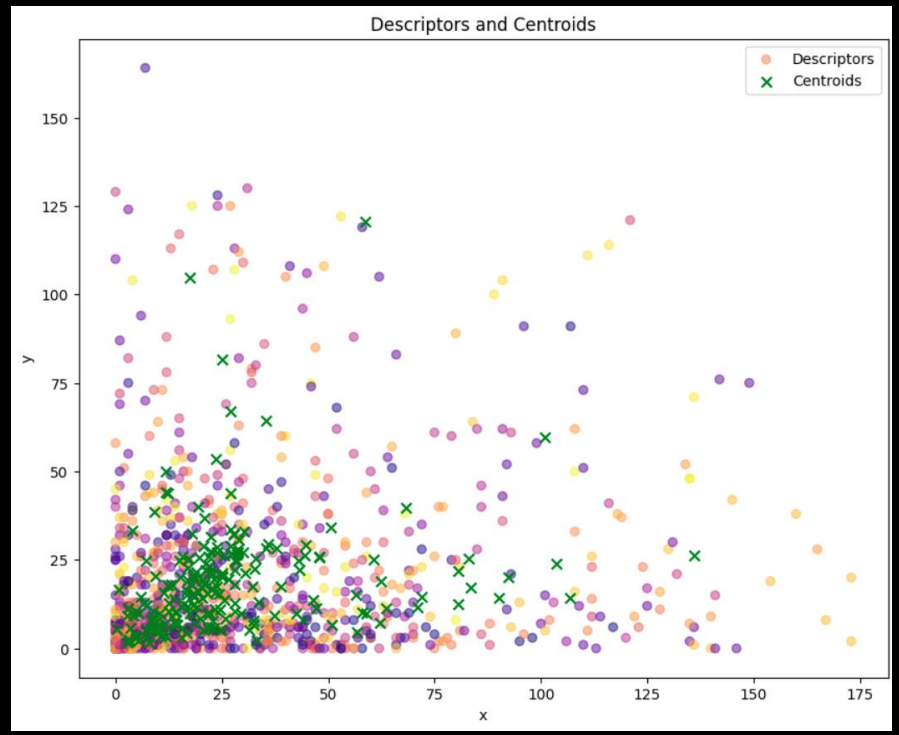


•Bag Of Features (BoF) is employed to create a vocabulary of features from training data

• BoF aggregates features into common bags, enabling subsequent algorithms to better capture of non-linear relationships.

•K Means clustering partitions the feature space, forming centroids that define words in the vocabulary. (as seen in the scatter plot on the right)

•Feature histograms are formed based on nearest centroid assignment, mapping long vectors to fixed-length outputs, addressing high dimensionality. Parameter K, determining the number of centroids, requires tuning. After iterations, 200 centroids were chosen to balance detail and prevent underfitting.



RESULTS

How are we evaluating performance?

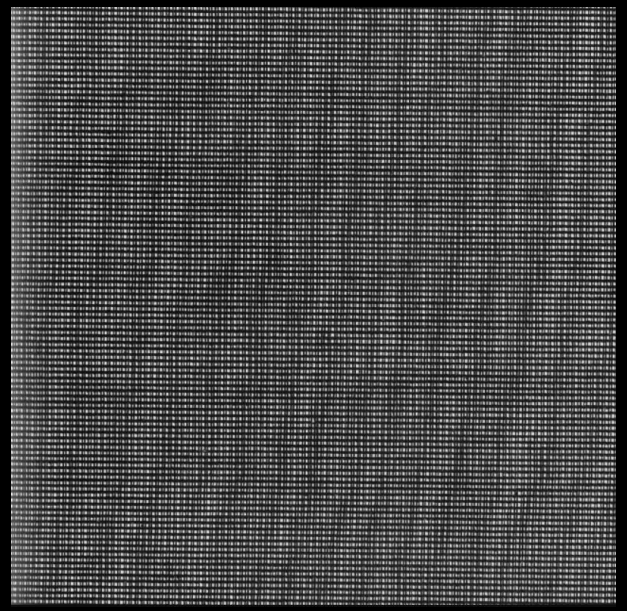
- Statistical metrics i.e.: accuracy, precision, recall
- Visualising model fit to data
- Investigating how model performances on augmented inputs
- Time considerations

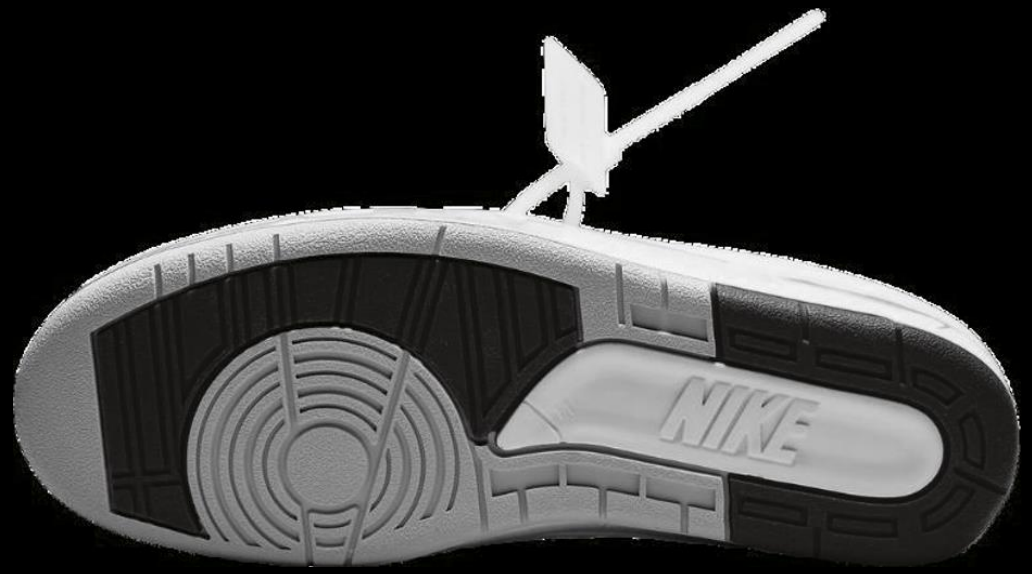
Methods	N = 1	N = 3	N = 5
LBP	44.5%	63.6%	72%
SIFT	66.5%	68.4%	71%
LBP-CNN	28.6%	48%	67%
SIFT-Conv1D	65.3%	88.7%	92.3%
SIFT-Conv2D	53.2%	76.2%	87%
CNN	67.5%	85.5%	93.4.0%
Pretrained-CNN	**95%**	**96.5%**	**98%**

Table 1: Top-N accuracy for the different classification methods. As we increase N, we increase the set from which we consider whether the correct classification has been considered

LBP

- Benchmark **LBP** performance on Broadtz dataset showed 75% accuracy, lower than the reported 91.2%.
- LBP descriptors exhibited poor classification performance (44.46% accuracy) due to specialization in finer texture elements, unsuitable for images with large homogeneous regions.

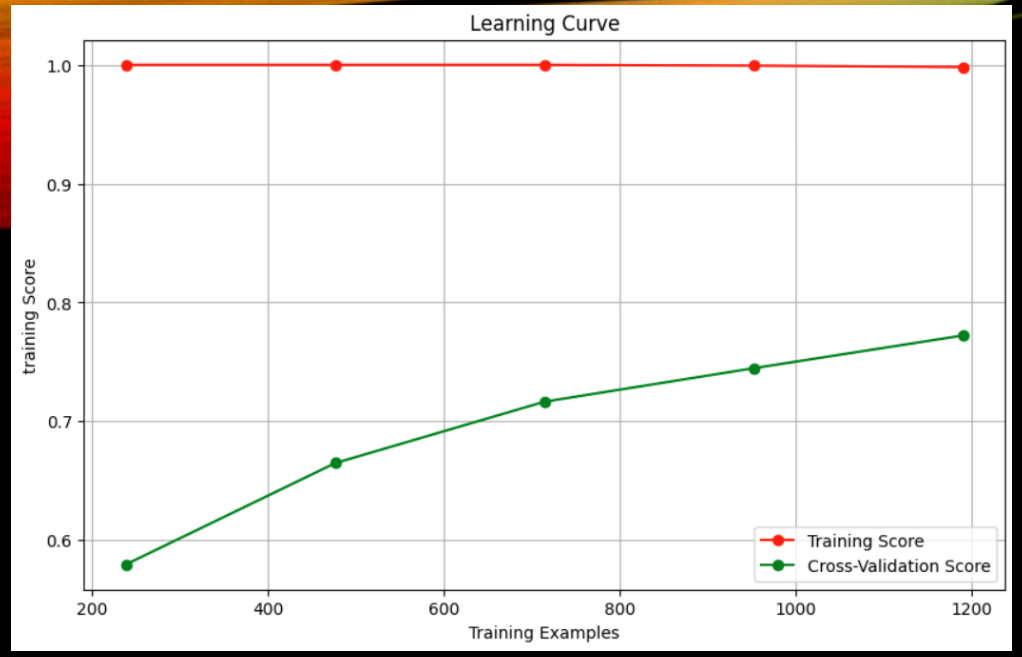




SIFT

- **SIFT** descriptor outperforms LBP with an accuracy of 66.5% on normal dataset and augmentations, maintaining consistent accuracy across different augmented images.
- Eroded images exhibit lower detection rates due to significant information loss during erosion augmentation. Implementation of erosion augmentation introduces new local gradient information, impacting keypoint selection in SIFT.
- SIFT entails higher computational costs, especially in matching, due to polynomial growth in matching times with dataset size. Matching algorithm using K-NN brute-force matcher is time-consuming, with a complexity of O(mnd) * O(mn log n), being impractical for the entire dataset, necessitating the use of a subset for efficient testing.

SIFT SVM



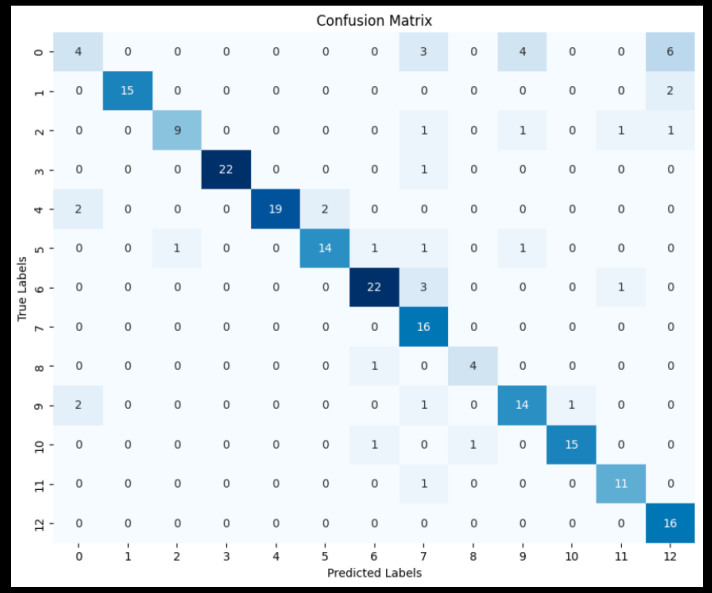

```
optimal params :  {'C': 10, 'degree': 3}
optimal achieved :  0.7876344086021505
---------- classification report -----------

              precision    recall  f1-score   support

           1       0.70      0.70      0.70        23
          10       0.94      0.73      0.82        22
          11       0.68      0.87      0.76        15
          12       0.74      0.77      0.75        26
          13       0.79      0.83      0.81        18
          14       0.70      0.75      0.72        40
           2       0.71      0.74      0.72        23
           3       0.66      0.90      0.76        21
           4       0.94      0.74      0.83        42
           5       0.86      0.89      0.87        35
           6       0.79      0.84      0.81        31
           7       0.81      0.85      0.83        34
           8       0.91      0.71      0.80        42

    accuracy                           0.79       372
   macro avg       0.79      0.79      0.78       372
weighted avg       0.80      0.79      0.79       372

------- Confusion Matrix: -------------

[[16  0  0  0  1  3  1  0  0  1  0  0  1]
 [ 2 16  0  1  0  1  1  0  0  0  0  1  0]
 [ 0  0 13  1  0  1  0  0  0  0  0  0  0]
```

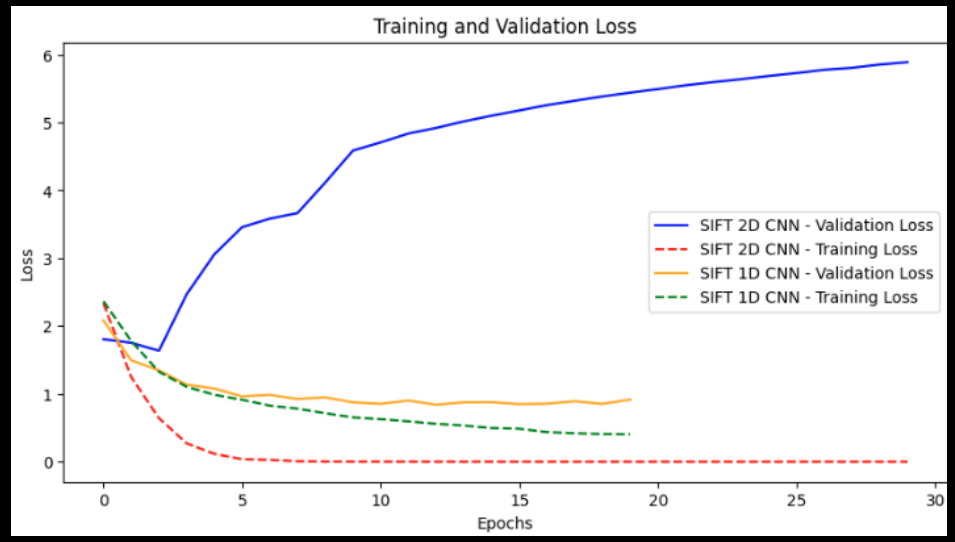
- SVM learning curve demonstrates high training score regardless of dataset size, with cross-validation score converging towards training score. Continued improvement in cross-validation score indicates potential performance enhancement with additional data.
- SIFT-SVM achieves promising accuracy of 0.78 with optimal configuration obtained through grid search: poly kernel with degree 3 and regularisation coefficient of 10. Consistent F1, precision, and recall scores (0.79, 0.80, and 0.79 respectively) indicate successful boundary identification for effective class separation. Cross-validation with 5 folds confirms model's ability to generalize, yielding consistent results across different unseen datasets (0.79 ± 0.01).

SIFT CNN PERFORMANCE (1D VS 2D)

Classification Report:

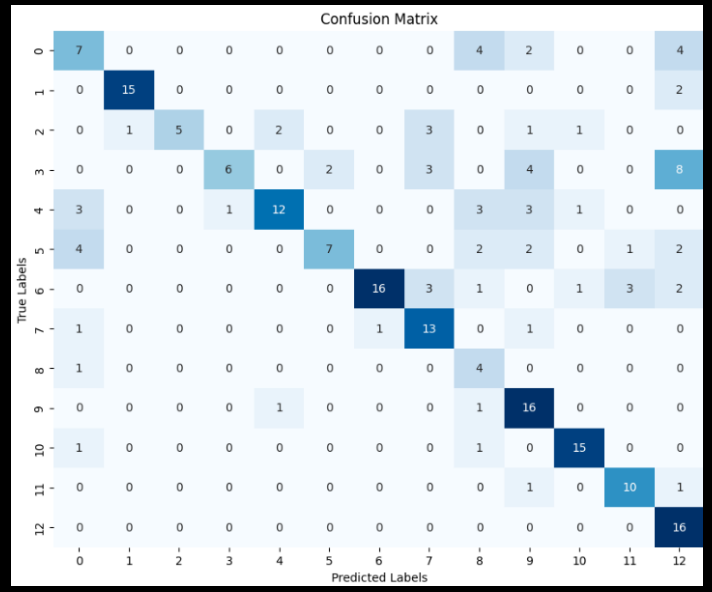
	precision	recall	f1-score	support
0	0.67	0.61	0.64	23
1	0.62	0.68	0.65	22
2	0.47	0.53	0.50	15
3	0.58	0.73	0.64	26
4	0.73	0.61	0.67	18
5	0.59	0.85	0.69	40
6	0.74	0.61	0.67	23
7	0.85	0.81	0.83	21
8	0.84	0.76	0.80	42
9	0.73	0.91	0.81	35
10	0.79	0.71	0.75	31
11	0.80	0.71	0.75	34
12	0.96	0.57	0.72	42
accuracy			0.72	372
macro avg	0.72	0.70	0.70	372
weighted avg	0.74	0.72	0.72	372

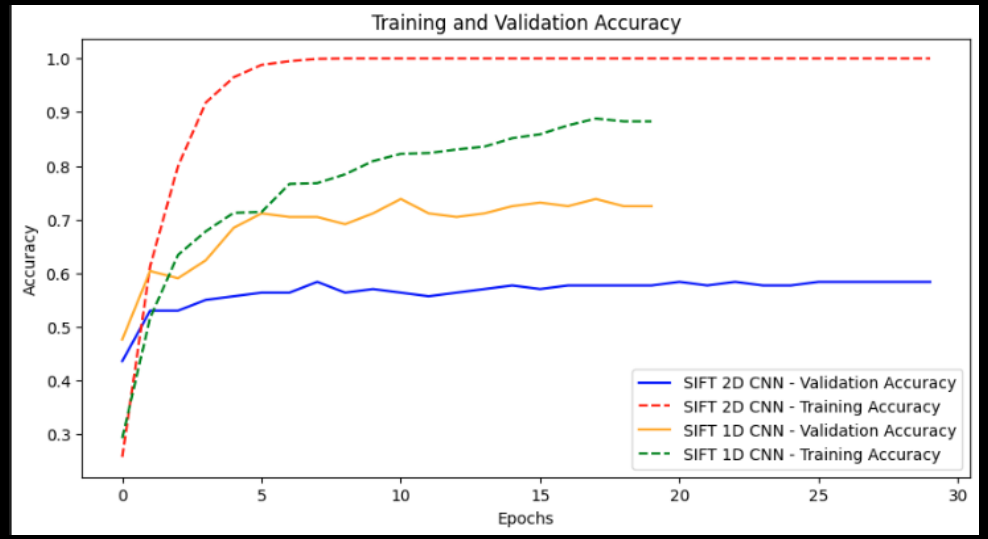




Classification Report:

	precision	recall	f1-score	support
0	0.38	0.57	0.46	23
1	0.62	0.59	0.60	22
2	0.44	0.27	0.33	15
3	0.50	0.42	0.46	26
4	0.41	0.39	0.40	18
5	0.55	0.40	0.46	40
6	0.48	0.57	0.52	23
7	0.38	0.76	0.51	21
8	0.63	0.52	0.57	42
9	0.53	0.60	0.56	35
10	0.62	0.48	0.55	31
11	0.69	0.65	0.67	34
12	0.62	0.60	0.61	42
accuracy			0.53	372
macro avg	0.53	0.52	0.52	372
weighted avg	0.55	0.53	0.53	372





PRETRAINED CNN ON IMAGE DATA

Why did it perform so well?

- Contrasting the SIFT-CNN(2d), pre-trained weights very well suited to the images in our dataset
- ImageNet encapsulates a vast amount of diversity across thousands of categories, and Is particularly useful in object recognition and scenery
- As such we able to take advantage of deeper networks without the training costs associated with them
- Vgg16 boasted best performance, likely due to model complexity where ResNet50 has a far deeper arrangement of layers

LINKS

- VIDEO LINK : DISS video - Made with Clipchamp.mp4
- CODE LINK: Shoeprint_dis.zip (one drive not accepting the uncompressed folder)