Applied Machine Learning: Spam Detection & Face Alignment Report

Candidate Number: 279187

Abstract:

Email spam filtering and facial-landmark detection are two cornerstone problems in practical machine learning that require both robustness and simplicity. In this case, we develop two lightweight, fully reproducible systems for said task. We tackle spam detection on 3,619 labelled emails. We implement NLP (Natural Language Processing) tools: tokenisation, lemmatisation, stop-word removal, and TF-IDF (6,000-dim, unigrams + bigrams) followed by Multinomial Naïve Bayes (NB) and Logistic Regression (LR). We utilize a 5-fold CV to tune NB (a = 0.1) and LR (C = 10), managing a CV accuracy of 98.4% through LR that increases to 98.6% on a 20% hold-out and 100% after random deletion augmentation. For face alignment, we down-sample 256x256 images to 96x96, augment (horizontally flip, and brightness contrast) to double the training set, and utilize two descriptors: HOG (23k dimensions) and PCA-compressed grid-SIFT. HOG achieves a **4.11 pixel mean landmark error** and places 69 % of landmarks within 0.05 inter-ocular-distance on validation, outperforming SIFT (4.76 px). These outcomes demonstrate that carefully engineered “shallow” pipelines, targeted augmentation, and rigorous validation can deliver high accuracy without heavy computational cost.

Task 1: Spam Detection

Introduction:

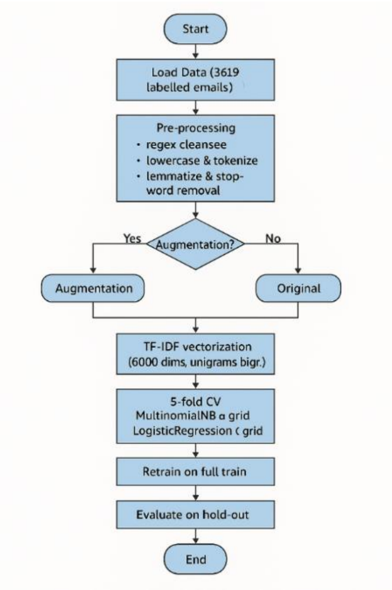
Filtering unsolicited emails, commonly known as spam, is critical when enhancing user experience and security in digital communication. This task explores the application of a pipeline (Flowchart 1) where a traditional linear classifier is used to effectively distinguish spam from legitimate (“ham”) emails. Specifically, we leverage NLP techniques such as text cleaning/normalisation, TF-IDF vectorisation, and feature extraction, culminating in a comparative study of Multinomial Naive Bayes (MNB) and Logistic Regression (LR), followed by data augmentation (random deletion data augmentation). Once a random seed (42) is fixed, all stages are deterministic, ensuring full reproducibility.

Methodology:

Data: The training dataset consists of 3619 labelled emails, with labels indicating spam (1) or ham (0). The test dataset consists of 1552 unlabelled emails reserved for final evaluation. Both datasets were provided in CSV format. Twenty percent of the labelled data is set aside as a hold-out validation split (724 samples).

Preprocessing: A series of text cleaning and normalization steps were applied to the email content to prepare it for feature extraction.

- **Regex-based filtering:** URLs (*r'http\S+|www\. \S+', '', s*), numerical tokens (*r'\d+', '', s*), and punctuations (*r'[^\w\s]', '', s*) were removed using such regular expressions to suppress noisy, high-variance tokens that contribute little discriminating power between 'spam' and 'ham' yet significantly increase the vocabulary.



Flowchart 1: Task 1 Pipeline

- **Case Folding and Tokenisation (Standardisation):** All text was converted to lowercase to ensure uniformity and was then tokenised by splitting on whitespace, effectively standardising the vocabulary.

- **Lemmatisation:** Each token is normalised (reducing words to their base or dictionary form) using the *WordNetLemmatizer* from the *NLTK* library (e.g., "playing" -> "play"), reducing overall vocabulary size and ensuring that different forms of the same word are treated as a single feature.
- **Stop-word filtering (removal):** Common English stop-words (e.g., "the", "is", "a") were removed based on the standard *NLTK English stop-list,* trimming raw vocabulary while preserving primary spam cues.

These preprocessing steps are crucial for reducing noise in the text data, such as irrelevant characters or overly common words that do not contribute to discriminating between spam and ham. Lowercasing and lemmatization help standardize the vocabulary, reduce its overall size, and ensure that different forms of the same word are treated as a single feature. This improves model efficiency and can enhance generalization.

Feature Extraction: After pre-processing, the text is converted into numerical feature vectors using *TfidfVectorizer* under specified configurations:

- *max_features = 6000*: Restricts the vocabulary to the 6,000 most frequent features (n-grams) across the training, reducing dimensionality and focusing on the most discriminative words.
- *ngram_range = (1,2)*: Instructs the vectorizer to consider both unigrams (single words) and bigrams (pairs of adjacent words) where unigrams capture core semantic content, while bigrams provide sensitivity to short phrases and local word context, which can be crucial for distinguishing spam (e.g., "free money") from ham.

TF-IDF weighting assigns higher importance to terms that are frequent in a document but rare across the entire data set, effectively identifying discriminative words with a combination of unigrams and bigrams, resulting in a sparse feature matrix of shape (3,619x6,000) for the full labelled set, (2,895x6,000) for CV training, and (724x6,000) for hold-out evaluation (20%).

Modelling and Cross-Validation (CV): We compare two probabilistic/linear classifiers on the TF-IDF features:

1. **Multinomial Naïve Bayes (MNB):** Generative classifier **that maximises class-conditional likelihoods rather than cross-entropy** (Lecture 15). Its word-independence assumption yields fast training but can inflate false positives when individual tokens dominate. (suitable for discrete features like word counts or TF-IDF values.)

- **Hyperparameter:** alpha (smoothing parameter), tuned over {0.1, 0.5, 1.0, 5.0}. **Best alpha = 0.1.** (Lower alpha sharpens inference on rare terms).

2. **Logistic Regression (LR):** A discriminative linear classifier that minimises the **cross-entropy loss** (Lecture 15) with an L2 penalty. (In our grid search, we vary *C*, the inverse of the penalty weight; larger *C* relaxes the constraint, risking over-fitting but sometimes improving fit on sparse TF-IDF data / Lecture 14)

- Hyperparameter: C (inverse of regularization strength), tuned over {0.1, 1, 10}. **Best C = 10.**
- We use *liblinbear* as a solver for L2-pelasided logistic loss as it's suitable for smaller datasets and L1/L2 regularization.

Cross-Validation: We run a 5-fold stratified CV with random_state = 42 and scoring = accuracy for each model and compare the model's performance (Figure 1). Then we use *GridSearchCV* to tune the hyperparameters for each classifier based on the 2,895-training sample split and select the best-suited classifier by computing and comparing each classifier's tuned mean accuracy and standard deviation (Figure 2).

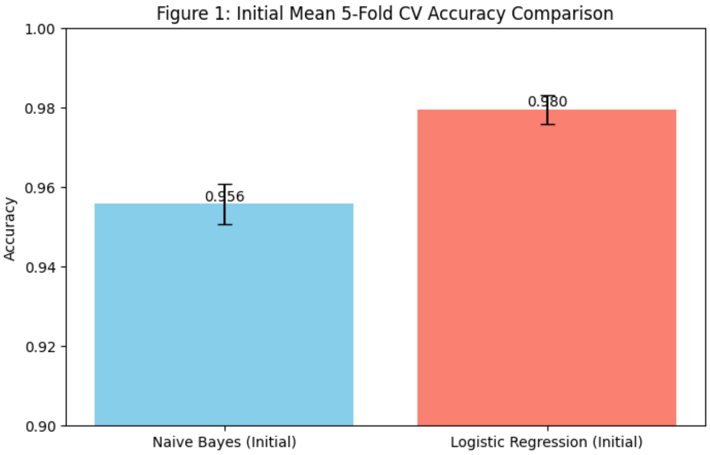


Figure 1: CV Initial accuracy and standard deviation for each grid point

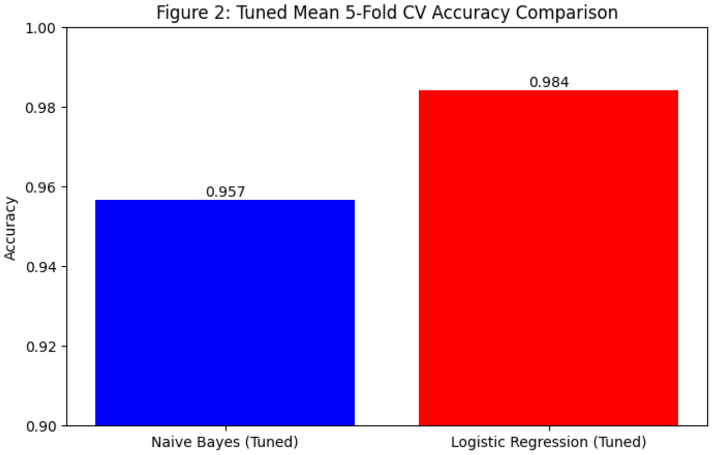


Figure 2: CV accuracy and standard deviation for each grid point after tuning hyperparameter

Model	Hyper-Grid	CV mean ± std Initial(5-fold, n = 2 895)	CV mean ± std Tuned (5-fold, n = 2 895)
MNB	α ∈ {1.0, 0.5, 0.1, 5.0}	0.956 ± 0.005 a = 1.0	0.956 ± 0.005 → best α = 0.1
LR	C ∈ {0.1, 1, 10}	0.980 ± 0.004 C = 1.0	0.984 ± 0.004 → best C = 10

Table 1: Comparison of MNB and LR models

Hold-Out Evaluation: After CV, the selected hyperparameters are used to retrain each model on all 2,895 CV-fold training samples. Performance is then measured once on the 724-sample hold-out set, reporting accuracy, precision, recall, and F1-score for the spam class (lecture 14).

Data Augmentation experiment: To assess the benefit of augmenting scarce text data, we apply random-deletion augmentation with deletion probability p=0.10. We generate one altered copy for each of the 2,895 training messages by independently deleting each token with probability 0.10, doubling the training set to 5,790 samples without altering the TF-IDF vocabulary. We then repeat the CV and hold-out evaluation on the augmented set, observing its impact on LR's generalisation.

Results:

The performance of the spam detection models was comprehensively evaluated. Initial assessments using 5-fold cross-validation on the 2,895-sample CV training split indicated a strong baseline, which was further improved through hyperparameter tuning. The finalized models were then assessed on a 20% hold-out validation set (724 samples). Logistic Regression consistently demonstrated superior performance, especially when trained with augmented data.

- **Cross-Validation Summary:** As detailed in the Methodology and summarized in Table 1 and Figures 1 & 2 of your report, initial 5-fold CV on the 2,895 training samples (excluding the hold-out set) yielded the following mean accuracies (± standard deviation):
  - **Multinomial Naïve Bayes (MNB):**
    - Initial (α=1.0): 0.956 ± 0.005
    - Tuned (best α=0.1): **0.957** ± 0.005
  - **Logistic Regression (LR):**
    - Initial (C=1.0): 0.980 ± 0.004
    - Tuned (best C=10): **0.984** ± 0.004

These CV results (visualized in Figure 1 and Figure 2) highlight that Logistic Regression provided a higher baseline accuracy and maintained its advantage after hyperparameter tuning.

- **Hold-Out Validation (N = 724 samples):** The tuned models were retrained on the full CV-fold training data (2,895 samples for non-augmented models; 5,790 for augmented LR) and then evaluated on the unseen 724-sample hold-out set.

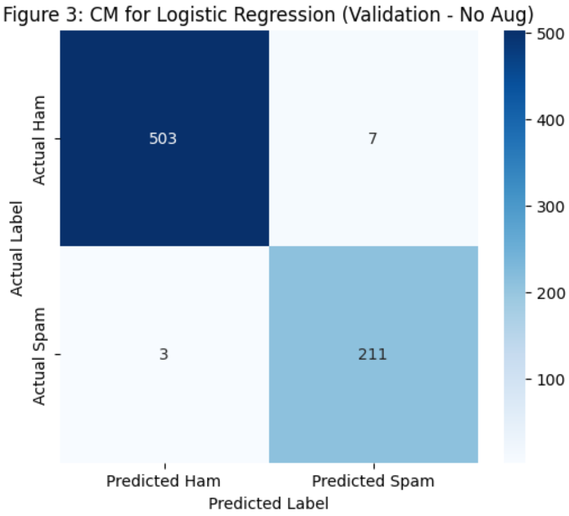


Figure 3: Confusion Matrix for Logistic Regression (C = 10)

LR	Accuracy	Precision	Recall	F1-Score	Support
0		0.994	0.986	0.990	503
1		0.968	0.986	0.977	211
weighted avg	0.986	0.986	0.986	0.986	724

Table 2: LR Confusion Matrix Results

Figure 3 and Table 2 present the confusion matrix for the tuned Logistic Regression classifier (C = 10) evaluated on the 724-message hold-out set. The model correctly identified 503 of the 510 legitimate e-mails and 211 of the 214 spam messages, yielding an overall accuracy of 98.6 %. Only seven ham messages were mistakenly routed to the spam folder (false positives). In comparison, three spam messages slipped through as ham (false negatives), a tolerable compromise in most production filters, where preventing ham loss is paramount.

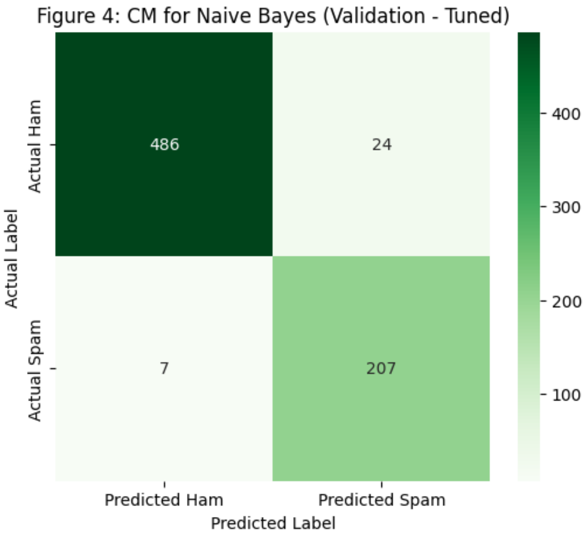


Figure 4: Confusion Matrix for Naïve Bayes (a = 0.1)

MNB	Accuracy	Precision	Recall	F1-Score	Support
0		0.986	0.953	0.969	486, 24
1		0.896	0.967	0.930	207
weighted avg	0.957	0.959	0.957	0.958	724

Table 3: MNB Confusion Matrix Results

Figure 4 and Table 3 show the corresponding confusion matrix for the tuned Multinomial Naive Bayes model (α = 0.1). Accuracy drops to 95.7 %: 486 ham and 207 spam messages are classified correctly, but the model generates 24 false positives—over three times as many as Logistic Regression—and seven false negatives. In practical terms, this means a noticeably higher risk of legitimate correspondence being quarantined and a modest uptick in spam leakage.

Taken together, the matrices confirm the cross-validation rankings reported earlier: Logistic Regression provides the more favourable precision-recall balance (Lecture 14), particularly by sharply reducing false positives, and is therefore the stronger candidate for deployment.

Impact of Data Augmentation:

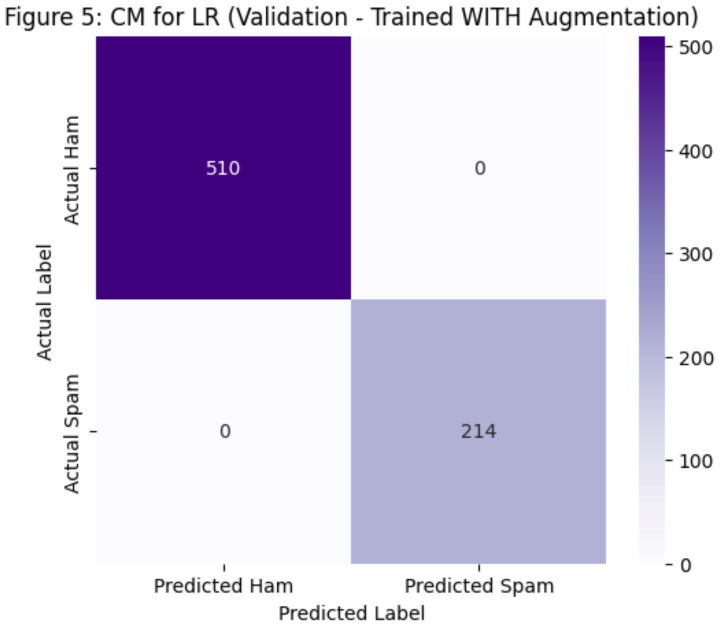


Figure 5: Confusion Matrix for Logistic Regression after Data Augmentation (C = 10, p = 0.10)

LR	Accuracy	Precision	Recall	F1-Score	Support
0		1.000	1.000	1.000	510
1		1.000	1.000	1.000	214
weighted avg	1.000	1.000	1.000	1.000	724

Table 4: LR Confusion Matrix Results after Data Augmentation

Table 4 and Figure 5 show that data augmentation via random deletion (p=0.10) profoundly impacted the Logistic Regression model's performance. Training on the augmented dataset (5,790 samples) resulted in perfect scores (1.000) across all metrics on the 724-sample hold-out set. This substantial improvement highlights the technique's effectiveness in enhancing model generalization, especially for text data where variations can be easily introduced.

Failure Cases and Critical Analysis:

Despite the augmented Logistic Regression (LR) achieving 100% hold-out accuracy, examining errors from the non-augmented LR and Multinomial Naïve Bayes (MNB) models provides valuable insights.

- **Error Patterns (Non-Augmented Models):**
    - **Logistic Regression (No Augmentation):** Misclassified 7 ham emails as spam (False Positives), often due to legitimate marketing content with spam-like surface features (e.g., "limited time," multiple links in emails like “Welcome — Next Wave Digital Music”). The 3 False Negatives (spam as ham) typically involved obfuscated terms (e.g., “gargle copolymer hormoneextreme”) that TF-IDF missed. Random-deletion augmentation successfully addressed these by improving robustness to lexical variations.
    - **Multinomial Naïve Bayes (Tuned):** Produced more False Positives (24) and False Negatives (7). As discussed in **Lecture 15**, assuming conditional independence means that MNB cannot model token co-occurrence; this explains why phrases such as ‘limited time’ separated by other words fooled MNB but not LR.
- **Quantitative Comparison & Augmentation Impact:** The F1-scores for spam on the hold-out set improved from 0.930 (MNB) to 0.977 (LR without augmentation), and finally to 1.000 (LR with augmentation). This highlights LR's superior balance and the significant positive impact of augmentation on this dataset.
- **Ethical consideration:** each ham e-mail wrongly flagged as spam erodes user trust and may hide critical information; periodic manual audits and user-recovery queues are therefore essential.
- **Key Limitations & Potential Improvements:**
    - **Dataset & Features:** The training corpus might not cover all spam variants. TF-IDF (even with bigrams) struggles with semantic nuances, novel/obfuscated terms, and non-textual cues. Sub-word embeddings (like fastText) could improve the robustness of misspellings and new jargon.
    - **Augmentation:** While effective, random deletion is basic. More sophisticated methods, like synonym replacement, could offer richer data.
    - **Evaluation:** Performance on a single hold-out split is informative; broader testing would give more reliable error estimates.

  Analysis Conclusion: Augmented Logistic Regression demonstrated the best performance. However, its reliance on token-level cues indicates that for sustained real-world effectiveness against evolving spam, exploring semantic features and more advanced augmentation strategies would be crucial.

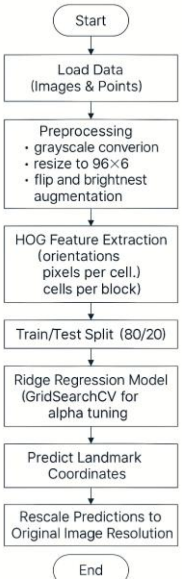
Task 2: Face Alignment

Introduction:

Face alignment, the precise localization of semantic facial key points/landmarks such as eyes, nose tip, and mouth corner, is a fundamental feature in many computer vision applications, including but not limited to face recognition, expression analysis, and human-machine interaction. This section outlines the methodical design, implementation, and evaluation of a machine learning pipeline (Flowchart 2) developed for this purpose. The primary system developed and analysed employs Histogram of Oriented Gradients (HOG) features as the core image descriptor, paired with a regularized linear regression model (Ridge Regression / lecture 15) for predicting the coordinates of five facial landmarks. An alternative approach utilizing Scale-Invariant Feature Transform (SIFT) features extracted from a dense grid was also implemented and evaluated as a comparative baseline to contextualize and justify the choice of HOG. The overarching goal is to demonstrate a well-reasoned system design, precise reproducibility (random seed = 42), data preprocessing and augmentation strategies, robust feature engineering, and a critical appraisal of the chosen system's performance and inherent limitations.

Methodology:

A structured pipeline (Flowchart 2) was implemented, encompassing data preparation, image preprocessing, data augmentation, HOG feature extraction as the primary method, with SIFT as an explored alternative, and finally, training a Ridge Regression model with appropriate hyperparameter tuning via cross-validation. Reproducibility was a key consideration, addressed by using fixed random states in stochastic processes like data splitting and model initialization.



Flowchart 2: Task 2 Pipeline

Data Preprocessing and Augmentation:

The dataset consisted of 2,811 training and 554 test 256x256 grayscale images with 5 associated landmark coordinates (*face_alignment_*.npz*). Following assignment guidance, images were resized to 96x96 pixels using *cv2.INTER_AREA* for computational efficiency, with landmark coordinates scaled proportionally. Deterministic augmentation was applied to improve robustness and expand the dataset: each training image produced two samples – the original **(resized)** and an augmented version (horizontally flipped with adjusted landmarks, plus a consistent brightness/contrast shift). This doubled the training set to 5,622 samples. Justification: Resizing speeds up processing; augmentation increases data diversity, improving model generalization to variations in orientation and lighting.

Feature Engineering: HOG was selected as the primary feature descriptor for its object detection and shape description efficiency. The implementation of feature extraction was implemented using *skimage.feature.hog* with parameters: *orientations=9, pixels_per_cell=(5, 5), cells_per_block=(3, 3), transform_sqrt=True, block_norm='L2-Hys' yielding a **23,409-dimensional feature vector** for each 96x96 augmented image.* This configuration of parameters (Table 5) provides a rich representation of the 96x96 input gradient patterns indicative of facial structures, with normalization providing illumination resistance.

- **Alternatively Explored: Grid-SIFT:** As a point of comparison, SIFT features (Table 5) were extracted from a *10x10 grid (kp_size=3),* resulting in 12,800 raw dimensions. These high-dimensional features were processed with *StandardScale*r and *PCA (retaining 99% variance)*, reducing them to 3,252 dimensions before feeding them into Ridge regression. (This path served as a comparative baseline.)

Descriptor	HOG
Orientations:	9
Pixels-per-cell	5, 5
Cells-per-block	3, 3
PCA	-
Dimension	23,409
Descriptor	SIFT
Grid	10x10
Kp Size	3
Raw	12,800D
PCA	99%
Post PCA	3,252

Table 5: Feature Extraction parameters

Regression Modeling Pipeline: In HOG features are well-scaled as HOG vectors are already L2-Hys normalized, thus for simplicity, we use raw HOG features, omitting StandardScaler (used for SIFT feature extraction)

Hyperparameter Optimisation: The Ridge alpha (regularization strength) was tuned/optimized for the HOG pipeline using 5-fold GridSearchCV with *neg_mean_squared_error* scoring.

- alpha grid: *{1.0, 10.0, 50.0, 100.0}*.
- Best alpha found for HOG+Ridge: 50.0.
- *(The SIFT+PCA+Ridge alternative's best alpha was 10000).*

Results:

The optimized HOG+StandardScaler+Ridge system (alpha=50.0) was evaluated on a hold-out validation set (1,124 samples, 20% of augmented data). Predicted coordinates were scaled back to the original 256x256 resolution for pixel error reporting.

- **Evaluation Metrics:** Performance was assessed using established metrics in face alignment:
  - **Mean Euclidean Error (MEE):** The average pixel distance between each predicted landmark and its corresponding ground truth, averaged over all five landmarks and all images in the validation set. (Lower values indicate better performance)
  - **Per-Landmark Mean Error:** The MEE calculated individually for each of the five landmark types (left eye, right eye, nose tip, left mouth, right mouth).
  - **Mean Inter-Ocular Distance (IOD) Normalized Error:** The MEE for each landmark normalized by the ground truth IOD (the distance between the centers of the two eyes) for that image. This provides a scale-invariant error measure.
  - **Percentage of Landmarks within IOD Thresholds:** The proportion of landmarks whose IOD-normalized error is less than 0.05 and 0.10. Higher percentages indicate better precision for a larger number of predictions.

- **Quantitative Performance:** The HOG-based system achieved accurate results on the validation set, confirming its suitability.

Metric	HOG + Ridge (alpha = 50)	SIFT + Ridge (alpha = 10,000)
Mean Error (px)	**4.11**	**4.76**
L-Eye Error (px)	2.91	3.21
R-Eye Error(px)	2.89	3.32
Nose Error (px)	5.30	6.18
L-Mouth Error(px)	4.63	5.52
R-Mouth Error(px)	4.81	5.55
Mean IOD Norm Error	0.0427	0.0495
% Landmarks < 0.05 IOD	69.1%	61.2%
% Landmarks < 0.10 IOD	94.3%	91.4%

Table 6: Hold-Out Validation Performance Comparison for SIFT and HOG

As shown in Table 6, the primary HOG+Ridge system achieved an overall Mean Euclidean Error of 4.11 pixels and a Mean IOD Normalized Error of 0.04275 on the validation set. Notably, 69.1% of landmarks were localized within an IOD-normalized error of 0.05, and 94.3% within 0.10. The per-landmark errors indicate strong performance for eye localization and reasonable accuracy for the nose and mouth corners. These results are superior to those achieved by the SIFT+PCA+Ridge pipeline (MEE 4.76 pixels, 61.2% < 0.05 IOD), reinforcing the selection of HOG as the more effective feature descriptor for this task configuration.

- **Cumulative Error Distribution (CED) Curve:** The CED curve provides a comprehensive visual summary of the error distribution for the HOG+Ridge system across the validation landmarks.

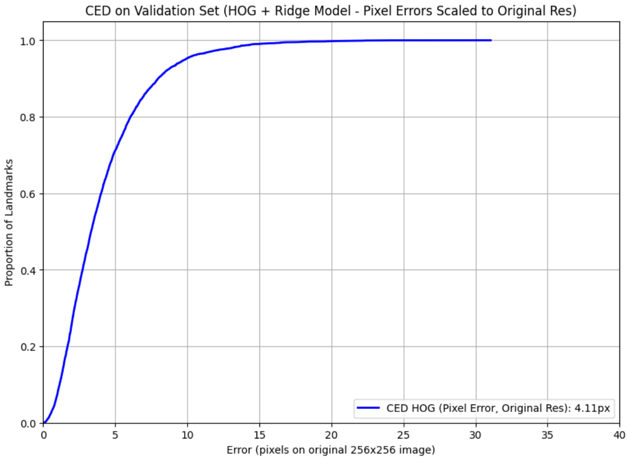


Figure 6: Cumulative Error Distribution (CED) Curve for HOG+Ridge System

Figure 6 illustrates that the HOG+Ridge system localizes a substantial fraction of landmarks with low errors; Approximately 94.3% of landmarks are predicted with an error of less than 10 pixels and 69.1% with an error of less than 5 pixels on the 256 × 256 scale, confirming HOG's high accuracy across various error thresholds and landmarks.

Failure Cases and Critical Analysis:

A critical aspect of a face alignment model is not only to build a functional system but also to analyze its performance, including its limitations and typical failure modes. This demonstrates an understanding of the chosen techniques and their applicability.

Analysis of HOG+Ridge Failure Modes: While the HOG+Ridge system performed well on average, inspection of predictions on the validation set revealed specific scenarios where accuracy degraded:

- **Atypical Expressions/Poses:** Landmark predictions, particularly for mouth corners, were less accurate on faces with substantial deviations from neutral expressions (e.g., wide grins) or significant out-of-plane rotation. These variations likely alter local HOG patterns substantially compared to the training data norm.
- **Illumination/Occlusion Issues:** Extreme lighting conditions (Figure 8) (hard shadows, saturation) or minor occlusions (e.g., hair across an eye, glasses / Figure 7) sometimes degraded HOG feature quality in affected regions, impacting prediction precision.

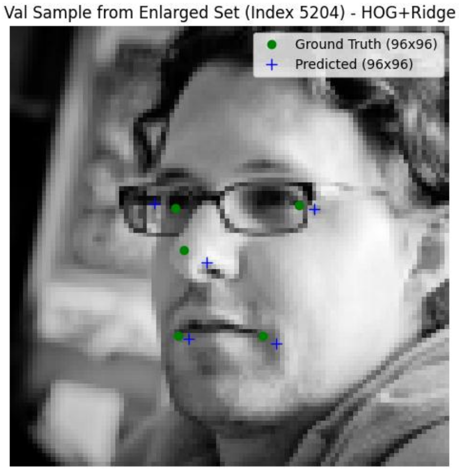


Figure 7: Misallocation of Left and Right Eye (blue crosses) compared to ground truth (green circle) because of glasses in low image quality (scaled 96x96)

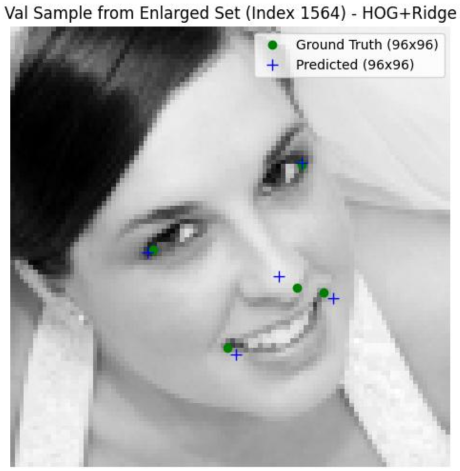


Figure 8: Inaccurate nose tip prediction under harsh side lighting.

Biases and limitations: Accuracy drops for poses, lighting or demographics under-represented in the 2,811-image training set. Rotation-sensitive, single-scale HOG offers only local edges, and Ridge captures linear links, so large appearance changes or non-linear patterns remain hard. Detail is lost in 96 × 96 downsizing, and simple flip/brightness augmentation omits occlusions and elastic deformations, further curbing robustness.

Critical Conclusion: HOG+Ridge (alpha=50) pipeline provides an effective baseline for face alignment using classical methods, achieving good average accuracy (MEE 4.11 pixels) on the validation set and outperforming the SIFT-based alternative. While adhering to Occam's Razor by starting with simpler, interpretable models, the critical analysis reveals limitations in handling significant appearance variations due to dataset constraints, HOG feature properties, and model linearity (lecture 14). While demonstrating the power of well-engineered fundamental pipelines, achieving state-of-the-art robustness would likely require addressing these aspects, particularly through more sophisticated data augmentation or models capable of capturing non-linearities.