Climbing Grade Classification on MoonBoard Using Hold Properties and Deep Learning

Juan Bello
Universidad Nacional de Colombia
Bogotá, Colombia
jdbelloro@unal.edu.co

***Abstract*—Climbing grade prediction aims to estimate the difficulty of bouldering problems based on route characteristics. While previous approaches have relied primarily on spatial information—such as the position of holds—this work explores whether incorporating hold-specific properties can improve grade classification. Using a dataset of approximately 60,000 MoonBoard problems collected from the official mobile app, we augment binary spatial encodings with detailed hold attributes to train a convolutional neural network. The goal is to evaluate whether this enriched representation leads to more accurate and consistent difficulty predictions compared to using spatial data alone.**

***Index Terms*—MoonBoard, climbing, deep learning, convolutional neural networks, climbing grades, spatial encoding**

I. Introduction

Climbing is a physically and technically demanding sport in which athletes ascend artificial or natural rock walls using a variety of predefined holds. On MoonBoard — a standardized 18×11 training wall — each route is defined by a fixed configuration of holds, and is shared globally through the MoonBoard app. Users assign difficulty ratings to problems based on personal perception, typically using the V-scale (e.g., V1 to V10), which originates from the Hueco Tanks bouldering system in the United States. Another common scale, particularly in Europe, is the Fontainebleau (or B-scale), which offers finer granularity at higher difficulty levels (e.g., 6A, 7B+, 8A).

Even among experienced climbers, substantial variability exists in estimating climbing difficulty based solely on visual inspection. For instance, Scarff [1] found that even highly skilled individuals correctly predicted the exact grade of MoonBoard problems in only about 45% of cases when relying solely on visual information. In broader populations of climbers, this accuracy can drop below 35% (GradeNet Project, 2023) [2]. However, when allowing a tolerance of ±1 grade—commonly referred to as one-off accuracy—performance improves notably, reaching approximately 85% among experts (Scarff, 2020). Despite this margin, over 60% of estimations among general climbers deviate by more than one grade (GradeNet Project, 2023), underscoring the inherently subjective nature of difficulty perception and the limited inter-rater reliability in visual grade assessments.

In MoonBoard setups, holds differ not only in spatial location but also in the way they are designed to be used. Unlike commercial gyms that include large slopers and volumes, MoonBoard emphasizes compact, high-difficulty holds. The 2016 MoonBoard configuration uses three official hold sets:

- **Set A (Yellow)**: Small, rounded, technical shapes mainly used for edge training.
- **Set B (White)**: A more varied collection including slopers, pinches, and edges. Suitable for versatile movement.
- **Set C (Black)**: Larger ergonomic holds designed for powerful or dynamic climbing. Often used for beginner-friendly problems.

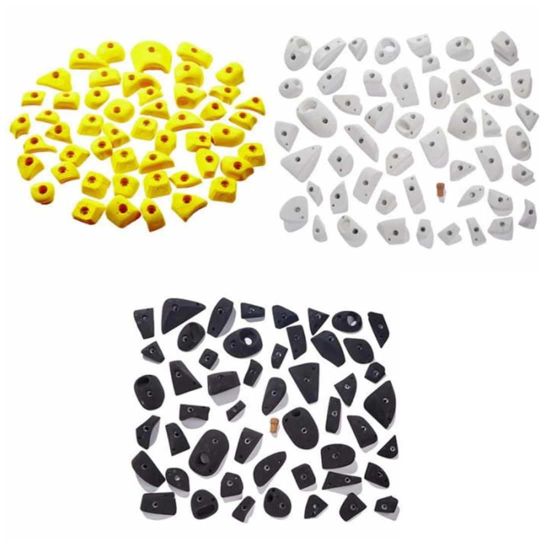
Fig. 1. MoonBoard Hold Sets (Image source: moonclimbing.com)

The difficulty of a climbing route is heavily influenced by the type of holds used and their orientation on the wall. Despite this, most deep learning approaches — such as the positiononly model by Petashvili and Rodda (2023) — focus solely on spatial location and ignore these semantic features.

The purpose of this work is to investigate whether incorporating semantic hold properties can significantly improve climbing grade prediction compared to using spatial configuration alone. Specifically, this study examines whether a convolutional neural network can better learn difficulty patterns when hold-specific attributes are added to the input representation. By comparing models trained with and without semantic features, this approach aims to evaluate the contribution and relevance of hold-level information in climbing grade estimation.

II. Related Work

Several approaches have been proposed for automated MoonBoard grade prediction and problem evaluation.

A. *Board-to-Board: Evaluating Moonboard Grade Prediction Generalization*

Petashvili and Rodda [3] explored the challenge of predicting climbing grades on MoonBoard while aiming for strong generalization across different board configurations. Their main goal was to develop models that could learn transferable features from one MoonBoard edition (e.g., 2016) and perform accurately on others (e.g., 2017, 2019). To this end, they evaluated several deep learning architectures, including CNNs trained on binary 18×11 matrices representing hold presence. They treated the problem as an ordinal regression task, mapping the grades to integers and minimizing mean absolute error (MAE). Their results showed that relatively simple CNNs, without access to movement sequences or user-specific data, could achieve competitive accuracy and generalize effectively. Additionally, they emphasized the role of benchmark problems for consistent evaluation and highlighted the benefit of using a clean, discrete grade encoding.

B. *Recurrent Neural Network for MoonBoard Climbing Route Classification and Generation*

Duh and Chang [4] introduced a system called *Betamove* designed to analyze and predict the difficulty of climbing problems by modeling movement sequences rather than static hold configurations. Their main objective was to understand what makes a climbing problem difficult by generating potential sequences of moves (or "betas") using a physics-informed graph traversal algorithm. These sequences are then evaluated with handcrafted heuristics that incorporate factors like body positioning, reachability, and move complexity. This approach diverges from image- or matrix-based methods by focusing on biomechanical feasibility and movement planning. While it does not rely on deep learning, Betamove offers valuable insight into the role of movement in climbing difficulty and serves as a complementary perspective to data-driven models.

C. *Learn to Climb by Seeing: Climbing Grade Classification with Computer Vision*

Garcia and Cárdenas [5] focused on developing a supervised learning model to predict the difficulty grade of MoonBoard climbing problems directly from route images. Their primary objective was to determine whether convolutional neural networks (CNNs) could extract sufficient features from static 2D images of MoonBoard problems to accurately estimate difficulty. Unlike prior works that relied on matrix representations or handcrafted features, their approach utilized screenshots of the board annotated with hold positions. They trained various CNN architectures on these images, comparing their performance and discussing how architectural depth and data augmentation affected accuracy. Although their model achieved moderate success, the study highlighted the limitations of visual input alone and suggested that integrating metadata or temporal information could improve performance.

III. Dataset

Due to the lack of a public API and limited direct access to problem metadata, the dataset was constructed using screenshots captured directly from the official MoonBoard mobile application. Each screenshot corresponds to a climbing problem from the 2016 MoonBoard configuration and includes the holds already highlighted, along with labels indicating the start and top holds, as well as the user-assigned V and B grades and the rating (1 to 5 stars).

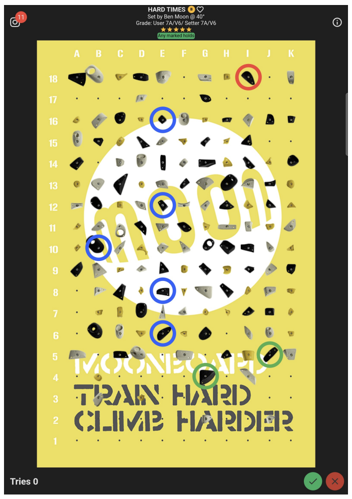
HARD TIMES
Set by Ben Moon @ 40°
Grade: User 7A/V6/ Setter 7A/V6
★★★★★
Any marked holds
Tries 0

Fig. 2. Visualization of a MoonBoard problem as displayed in the official app, with start and top holds highlighted.

A. *Route Distribution*

An important aspect of the dataset is the distribution of the target variable: the climbing grade.

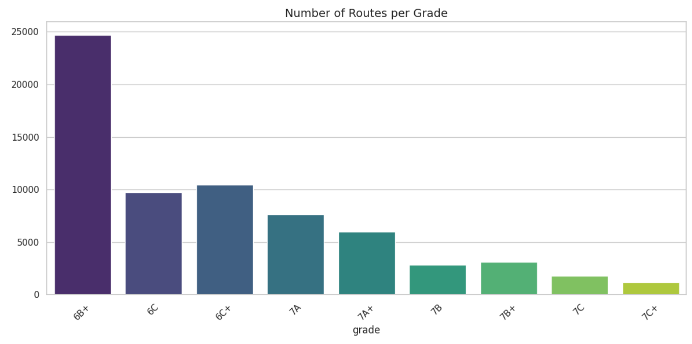

Fig. 3. Grade distribution.

- **Concentration in mid-grades (6B+, 6C, 6C+):** Most problems fall into this range, suggesting a natural tendency by both users and setters to create problems within lower-intermediate levels.

- **Long tail distribution:** There is a sharp decline in the number of problems as the grade increases, particularly beyond 7B+. This results in significant class imbalance.

B. Benchmark distribution

Benchmark problems are official MoonBoard climbs selected to provide consistent grading standards. They are carefully curated and used as reference problems for evaluating climber performance and calibrating grades. Despite their importance, they are extremely rare—only around 500 benchmark problems exist in a dataset of over 80,000 routes, representing just 0.6% of the total.

In terms of grade distribution, benchmark and non-benchmark routes exhibit distinct patterns:

- **Non-benchmark routes:**
  - These are by far the most abundant, with a strong skew toward moderate grades.
  - The most common grade is 6B+, followed by 6C+ and 7A.
  - The distribution shows a steep drop-off as difficulty increases beyond 7A+, with very few problems above 7C.
- **Benchmark routes:**
  - These are much fewer in number overall (note the difference in y-axis scale).
  - The distribution is more balanced, with a peak around 7A and 7A+.
  - Benchmark problems are more evenly spread between 6B+ and 7C+.
  - Even at higher grades (e.g., 7C+), benchmark representation remains consistent, indicating careful selection and validation.

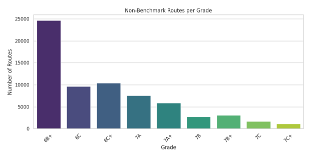

Fig. 4. Non Benchmark routes.

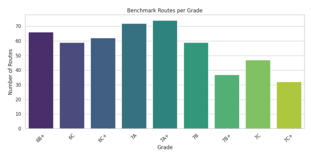

Fig. 5. Benchmark routes.

C. Hold Images

In addition to the route-level dataset, a separate collection of individual hold images was constructed from a photograph of the MoonBoard wall. These hold images were processed to generate embeddings that capture their visual features, with the aim of incorporating this semantic information into the models used for grade prediction.

IV. Data Collection and Preprocessing

To collect MoonBoard problems at scale, a custom scraping script was developed to automate screenshot capture from the official MoonBoard mobile application. The script simulates a user swiping through the problem feed, capturing each route image and saving it locally. The raw dataset contains 77,442 screenshots, of which 67,318 were retained after cleaning and validation.

For metadata extraction, the header section of each screenshot was cropped, and the board image was preserved as a separate file.

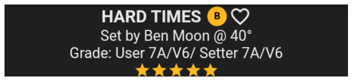
HARD TIMES B ♡
Set by Ben Moon @ 40°
Grade: User 7A/V6/ Setter 7A/V6
★★★★★

Fig. 6. Example of the cropped section which contains the route's metadata.

A. OCR

Optical character recognition (OCR) was applied to extract the grade with high reliability, owing to the consistent font and positioning used by the app interface. Given the size of the dataset, both the cropping and metadata extraction processes were parallelized using Python's *concurrent.futures* library, significantly reducing total preprocessing time.

B. Stars

Only problems with three or more visible star ratings were retained in the final dataset, under the assumption that higher-rated routes are more reliable in terms of community consensus on their difficulty grade. The number of stars was estimated through color-based image processing: a fixed rectangular region near the bottom of the header was extracted and converted to HSV color space. The region was divided into five equal-width sections corresponding to the possible star positions. For each section, the number of pixels falling within a defined yellow hue range was counted. If this count exceeded a set threshold, the corresponding star was considered filled.

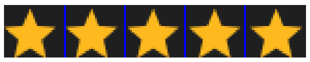

Fig. 7. Example of the cropped section which contains the stars.

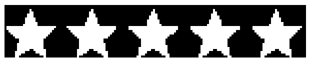

Fig. 8. Example of debug output showing detected star regions.

C. Benchmarks

Benchmark tags were initially identified by detecting yellow pixel density in a predefined area. However, this approach yielded unexpected false positives due to the presence of yellow emojis in route titles. To improve precision, a template-matching approach using normalized cross-correlation was implemented to detect the "B" benchmark badge with high confidence. A conservative threshold was adopted to avoid false positives, which led to the exclusion of 35 true benchmark routes. These instances were treated as false negatives and omitted from the benchmark subset.

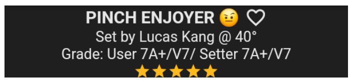
PINCH ENJOYER 🤨 ♡
Set by Lucas Kang @ 40°
Grade: User 7A+/V7/ Setter 7A+/V7
★★★★★

Fig. 9. Example of a route containing an emoji character in its name.

D. *Grade Standardization*

The grade column was standardized by removing irrelevant or inconsistent suffixes and correcting formatting anomalies. Initially, strings such as “Grade: User” were stripped, followed by the removal of trailing characters corresponding to alternative grading systems (e.g., “V5”) in order to retain only the French grading scale.

TABLE I
MOST FREQUENT INVALID GRADE FORMATS

Grade Format	Count
TAt	1355
S > & & & ¢	1171
TAIV6	1170
S a ao & aXe	1142
TA+	290
S a ao aXkaXd	133
6Bt	130
7TAIV6	87
7AIV6	39
TA	29
TAt+	14
TAIN6	10
TAIN	8
JA+	5
7TAIN6	4
unknown	2
TAT	1
vevarertare	1
TA+IVT	1
ee eS eed	1
7At	1

After the cleaning process, several malformed grade entries were identified, likely resulting from OCR errors or encoding issues. Examples included entries such as TAt, S > & & & ¢, and TAIV6. To address these anomalies, a semi-manual approach was adopted:

- The most frequent malformed formats were manually reviewed and corrected based on their context and associated route images, which consistently showed the same valid grade across all occurrences. This visual consensus was used as a reliable reference for assigning the correct label.
- Rare, ambiguous, or unidentifiable formats were discarded to maintain data integrity and avoid introducing noise.

This approach was used to retain the maximum number of valid samples while ensuring reliable and uniform grade labels for model evaluation.

E. *Grade Normalization*

The target variable (grade) was normalized to the $[0, 1]$ range using min-max scaling. This transformation is particularly useful for regression tasks in neural networks, where unbounded or unevenly scaled targets can lead to unstable gradients and slower learning.

Since MoonBoard grades are defined on an ordinal scale with a finite number of values, each valid grade (e.g., 6A, 6B+, 7A...) was first mapped to a corresponding integer index based on difficulty(0 to 9). The resulting ordinal encoding was then scaled linearly using:

$$\text{grade}_{\text{norm}} = \frac{\text{grade}_{\text{index}} - \min}{\max - \min}$$

where $\min$ and $\max$ correspond to the lowest and highest grade indices in the training set, respectively.

F. *Hold Image Preprocessing*

In addition to route-level information, a high-quality dataset of individual hold images was constructed to capture their visual properties. This process began with a high-resolution photograph taken of a physical MoonBoard setup. The image was divided into a grid corresponding to the 18×11 MoonBoard layout, and cropped images of each hold were generated. Since some holds extended beyond their grid boundaries or were misaligned, a second manual framing pass was performed to ensure accurate extraction.

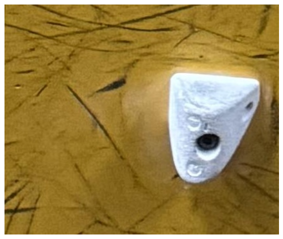

Fig. 10. Example of a cropped hold.

To isolate each hold from the background, the Segment Anything Model (SAM) by Meta AI was employed. Manual clicks on the target hold were used as visual feedback to guide the model in producing accurate segmentation masks. These masks enabled the extraction and centering of each hold on a uniform background. The segmented images were then padded and rescaled to a fixed size, preserving their relative proportions to ensure visual consistency across samples.

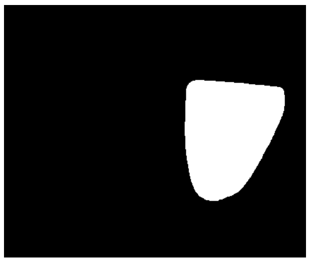

Fig. 11. Hold segmentation mask generated using the Segment Anything Model (SAM) developed by Meta.

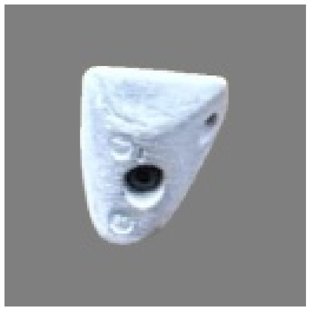

Fig. 12. Example of a hold image after preprocessing and resizing to 128×128 pixels.

V. DATA ENCODING

A. *Binary Matrix Encoding of Hold Positions*

All climbing problems were transformed from raw images into binary matrices aligned with the fixed MoonBoard grid of 18 rows and 11 columns. Each climbing problem was originally represented as a cropped image showing only the climbing wall, with holds visually marked in green (start holds), red (top hold), or blue (regular holds). While CNNs can operate directly on image data, this would incur unnecessary computational cost and introduce noise, since the background and graphical artifacts offer no added value for spatial learning.

Instead, a color-based detection was implemented to extract the coordinates of the marked holds and convert them into a compact matrix form. Each image was divided into a grid of 18 horizontal and 11 vertical segments. Color masks in HSV space were applied to detect green, red, and blue circles, corresponding to the hold annotations. These were then mapped to grid cells by locating the center of each detected contour. The final result is a binary matrix of shape 18×11, where each cell encodes the presence of a hold.

B. *Semantic Hold Embeddings*

Each individual hold image was processed and embedded using a convolutional autoencoder. The pipeline followed several key steps:

First, a mapping from hold IDs (e.g., "A5", "C10") to array indices was constructed to ensure each embedding could be correctly associated with its position in the 18×11 grid representation.

A **convolutional autoencoder** was then trained on this image dataset. The encoder compressed each image into a 32-dimensional embedding, corresponding to its representation in the latent space and capturing visual properties such as shape, size, and texture. During training, the decoder was used to reconstruct the original image from its latent representation, allowing the model to learn meaningful encodings. After training, only the encoder was retained to generate fixed semantic embeddings for each hold, which were later integrated into the route representations.

VI. Methodology

To ensure a fair comparison between models, the same training methodology and hyperparameters were applied uniformly across all architectures—except for the autoencoder, which was trained separately for the purpose of embedding generation. By standardizing the training protocol (including loss function, optimizer, batch size, early stopping, and learning rate), the influence of input representations and architectural variations was isolated.

A. *Model Architectures*

Convolutional Neural Networks (CNNs) and Long Short-Term Memory networks (LSTMs) were selected based on their superior performance in previous works, particularly in the study by Petashvili and Rodda [3], where they outperformed other baseline models in MoonBoard route difficulty prediction. These architectures were further chosen due to their complementary inductive biases, which align well with the structure of the input data.

CNNs were employed to capture the spatial configuration of holds within the MoonBoard grid. This is motivated by the observation that climbing routes exhibit strong spatial patterns, where the relative positioning of holds encodes useful information regarding difficulty. The ability of CNNs to exploit local connectivity and translation invariance enables them to learn such spatial relationships effectively, making them well-suited for grid-based representations of problems.

LSTMs, on the other hand, were used to model routes as ordered sequences of holds, mimicking the way climbers perceive and plan their movements—typically from bottom to top. By treating each row of the input matrix as a timestep, the LSTM was able to capture temporal dependencies and vertical progression within a route. This sequential modeling aligns with the cognitive process of climbers, who must navigate a route step-by-step, considering both past and upcoming moves.

Together, these two architectures were expected to provide robust baselines for evaluating the predictive value of different input representations.

An autoencoder was used to learn compact semantic representations of MoonBoard hold images through unsupervised training. Its purpose was not classification, but rather to extract visual features by reconstructing the input image. A convolutional architecture was selected for its ability to capture spatial hierarchies.

- **CNN**: Consisted of two 2D convolutional layers (32 and 64 filters), followed by batch normalization. After feature extraction, the output was flattened and passed through a dense layer with ReLU activation, followed by a final dense layer with linear activation for grade regression.

TABLE II
CNN Model Architecture

Layer (type)	Output Shape
InputLayer (18, 11, 1)	(None, 18, 11, 1)
Conv2D (32 filters, 3x3, ReLU)	(None, 18, 11, 32)
BatchNormalization	(None, 18, 11, 32)
Conv2D (32 filters, 3x3, ReLU)	(None, 18, 11, 32)
BatchNormalization	(None, 18, 11, 32)
Conv2D (64 filters, 3x3, ReLU)	(None, 18, 11, 64)
BatchNormalization	(None, 18, 11, 64)
Conv2D (64 filters, 3x3, ReLU)	(None, 18, 11, 64)
BatchNormalization	(None, 18, 11, 64)
Flatten	(None, 12672)
Dense (32, ReLU)	(None, 32)
Dense (1, Linear)	(None, 1)

- **LSTM**: Treated each route as a sequence of 18 rows with 11 features, reshaped from the input tensor and passed through a single LSTM layer (128 units), followed by a dense layer with 32 units and a final dense output layer for grade regression.

  For the baseline (spatial-only) approach, the input for both models was a binary matrix of shape $(18, 11, 1)$.

  In the semantic approach, the input was a tensor of shape (18, 11, 32), with each spatial position representing either the 32-dimensional visual embedding of a hold (if present in the binary mask), or a zero vector otherwise. This

TABLE III
LSTM Model Architecture

Layer (type)	Output Shape
InputLayer (18, 11, 1)	(None, 18, 11, 1)
Reshape	(None, 18, 11)
LSTM (128)	(None, 128)
Dense (32)	(None, 32)
Dense (output)	(None, 1)

allowed the model to preserve both spatial and semantic information."

- **Autoencoder**: Consisted of three convolutional layers (32, 64, and 128 filters) with max pooling in the encoder, followed by a dense embedding layer of size 32. The decoder mirrored the encoder using upsampling and convolutional layers (64, 32 filters) to reconstruct the original 128×128×3 hold image.

TABLE IV
Autoencoder Model Architecture

Layer (type)	Output Shape
InputLayer	(None, 128, 128, 3)
Conv2D (32)	(None, 128, 128, 32)
MaxPooling2D	(None, 64, 64, 32)
Conv2D (64)	(None, 64, 64, 64)
MaxPooling2D	(None, 32, 32, 64)
Conv2D (128)	(None, 32, 32, 128)
MaxPooling2D	(None, 16, 16, 128)
Flatten	(None, 32768)
Dense (embedding)	(None, 32)
Dense	(None, 32768)
Reshape	(None, 16, 16, 128)
UpSampling2D	(None, 32, 32, 128)
Conv2D (64)	(None, 32, 32, 64)
UpSampling2D	(None, 64, 64, 64)
Conv2D (32)	(None, 64, 64, 32)
UpSampling2D	(None, 128, 128, 32)
Conv2D (output)	(None, 128, 128, 3)

B. *Evaluation*

The evaluation was conducted using standard regression metrics: **Mean Absolute Error (MAE)** and **Root Mean Squared Error (RMSE)**. These metrics were chosen as they offer complementary perspectives on prediction quality: MAE measures the average magnitude of errors regardless of direction, while RMSE penalizes larger deviations more heavily.

Formally, the metrics are defined as:

$$\textbf{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|, \quad \textbf{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}$$

where $\hat{y}_i$ is the predicted grade (as a scalar ordinal value) and y_i is the real grade.

To qualitatively evaluate the structure of the learned embedding space, *t*-distributed Stochastic Neighbor Embedding (t-SNE) was applied to project the 32-dimensional hold embeddings into two dimensions. This technique was selected to assess whether semantically similar holds—those sharing characteristics such as shape, size, or texture—were mapped to nearby regions, thereby revealing coherent clusters in the latent space.

In this case, t-SNE was configured with the following parameters:

- **n_components** = 2: To obtain a 2D representation.
- **perplexity** = 15: To balance local and global structure.
- **learning_rate** = 200: For stable convergence.
- **random_state** = 42: To ensure reproducibility.

Subsequently, **K-Means clustering** was performed on the resulting 2D t-SNE embeddings using **k = 3**, reflecting prior knowledge that the MoonBoard system contains three official hold sets. This clustering was used to highlight emergent groupings and assess whether the visual embeddings captured structure aligned with these known categories.

C. *Benchmark-Based Split*

The model was trained using a total of 66,810 non-benchmark problems and evaluated on 508 benchmark problems, which were reserved as a held-out test set. This decision was motivated by the fact that benchmark problems in MoonBoard are curated and validated by the community, typically featuring high consensus on their assigned grade. As such, they represent a more reliable and stable evaluation target compared to user-submitted problems, which often suffer from noisy or inconsistent grading.

The split was performed in a stratified manner to preserve the grade distribution across both training and test sets. This ensures that the model is exposed to a representative range of difficulty levels during training and that the evaluation reflects realistic performance across all grades.

An earlier experiment was conducted using a random 80/20 split of the non-benchmark data for training and validation. However, the model was found to generalize better when trained on the full non-benchmark set and evaluated exclusively on benchmark problems.

D. *Training strategy*

Training was performed using the Adam optimizer and explored mean squared error (for regression on continuous grade representations) as loss function. Early stopping was applied with a patience of 20 epochs, monitoring the validation loss and restoring the best weights. To avoid premature termination, early stopping was configured to activate only after epoch 10.

VII. Results

A. *First approach: Spatial Relationships*

This experiment evaluates whether spatial information alone—i.e., the binary matrix of holds on the MoonBoard wall—is sufficient to predict climbing difficulty. To this end, two baseline models were implemented following the methodology proposed by Petashvili et al. (2023): a 2D Convolutional Neural Network (2DCNN) and a Long Short-Term Memory network (LSTM). Both models received as input a binary matrix of shape 18 $\times$ 11, indicating the presence or absence of holds.

The results, shown in Table V, indicate that both models perform similarly, with the LSTM slightly outperforming the CNN across both MAE and RMSE:

TABLE V
MAE AND RMSE ON BENCHMARK TEST SET

Model	MAE	RMSE
CNN	0.979	1.538
LSTM	0.948	1.355

To better understand the learning dynamics of each architecture, we analyzed their training curves, shown in Figures 13 and 14. Both models display signs of overfitting: the validation error diverges from the training error after fewer than 10 epochs. Moreover, the models appear to converge rapidly, reaching their lowest validation loss relatively early in training.

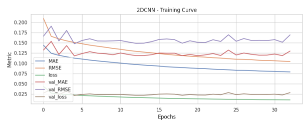

Fig. 13. Training and validation loss (MAE) for the CNN model using spatial input.

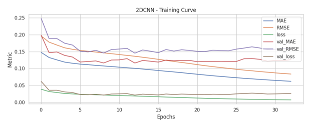

Fig. 14. Training and validation loss (MAE) for the LSTM model using spatial input.

B. *Hold Image Embeddings*

Beyond predictive performance, the structure of the embedding space was examined to assess whether the learned representations captured meaningful visual information about the holds. Since the embeddings were produced without explicit supervision, analyzing their organization provides insight into the model's ability to extract semantic patterns from visual data.

The resulting clusters, shown in Figure 15, revealed clear separation in the embedding space, suggesting that the learned representations effectively captured visual differences among holds.

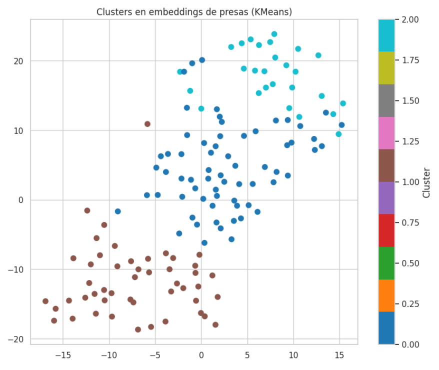

Fig. 15. Clusters in the 2D projection of the hold embedding space.

To assess the semantic consistency of the clusters, a set of randomly selected holds from each group was visualized. Each cluster exhibited a high degree of visual coherence, often grouping holds with similar geometric and textural characteristics.

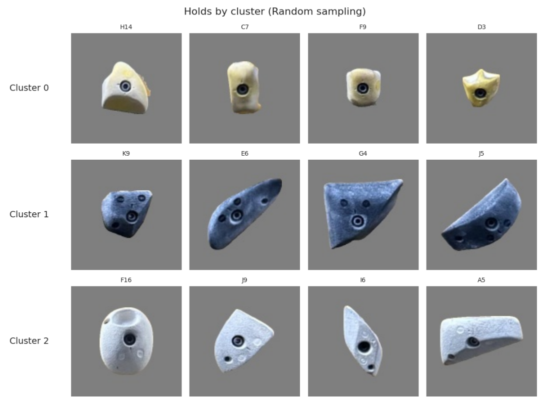

Fig. 16. Example holds from each cluster.

Interestingly, the discovered clusters closely aligned with the official MoonBoard hold sets.

C. *Second Approach: Introducing Hold Semantic Features*

In this second approach, the input representation was enriched with the semantic features extracted from hold image embeddings.

A mapping function was used to align each hold's embedding with its corresponding location in the MoonBoard grid, resulting in input tensors of shape $(18, 11, 32)$. These were then used to train two architectures:

- **2D CNN**, which applies convolutional filters over the enriched spatial-semantic grid.
- **LSTM**, which reshapes the input as a sequence of 18 steps (rows), each with 11×32 features, and models temporal dependencies via recurrent layers.

As shown in Table VI, the CNN model outperformed the LSTM, achieving the lowest error metrics across both evaluation measures.

TABLE VI
MODEL PERFORMANCE WITH SEMANTIC HOLD EMBEDDINGS

Model	MAE	RMSE
CNN	0.897	1.265
LSTM	0.931	1.359

Training dynamics for both models are shown in Figures 17 and 18. Similar to the spatial-only models, both architectures reached their minimum validation loss within the first 10–15 epochs. After this point, continued training led to increasing overfitting, as the training loss continued to decrease while the validation loss remained flat or worsened.

D. *Performance Comparison*

Figures 19 and 20 compare the performance of both CNN and LSTM architectures under two different input representations: spatial-only (baseline) and spatial enriched with

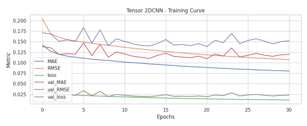

Fig. 17. Training and validation loss (MAE) for the CNN model (semantic input).

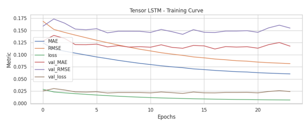

Fig. 18. Training and validation loss (MAE) for the LSTM model (semantic input).

TABLE VII
PERFORMANCE COMPARISON ACROSS ALL MODELS AND INPUT TYPES

Model (Input)	**MAE**	**RMSE**
CNN (Spatial-only)	0.979	1.538
LSTM (Spatial-only)	0.948	1.355
CNN (Semantic-enriched)	**0.897**	**1.265**
LSTM (Semantic-enriched)	0.931	1.359

semantic hold features. Across both evaluation metrics—MAE and RMSE—the inclusion of visual semantics improves the CNN model significantly, reducing both error values.

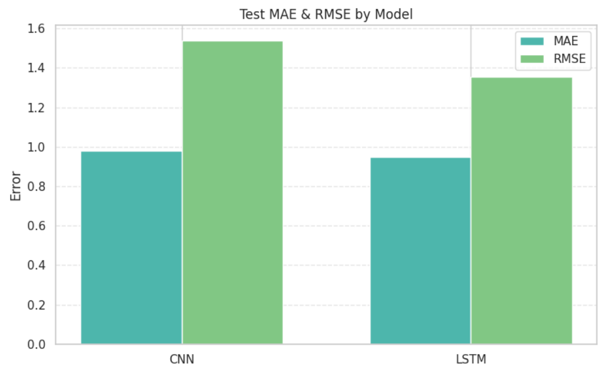

Fig. 19. Performance of CNN and LSTM using spatial-only input.

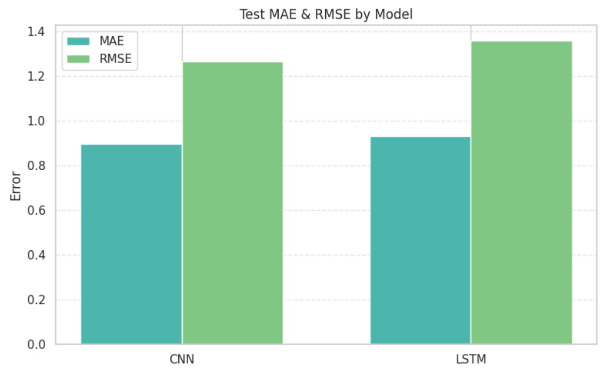

Fig. 20. Performance of CNN and LSTM using semantic-enriched input.

Although the LSTM model also showed slight improvement when using enriched input, the gain was less pronounced and its performance remained below that of the CNN. This suggests that while LSTMs can exploit row-wise structure, they may be less effective than convolutional filters in capturing local spatial-semantic correlations.

VIII. Discussion

A. *Summary of Findings*

This study investigated the impact of enriching MoonBoard route representations with visual semantics extracted from hold images. The experimental results confirmed that incorporating semantic features into the input—particularly for convolutional models—significantly improves prediction performance. CNNs trained on enriched input consistently achieved lower MAE and RMSE than their spatial-only counterparts, validating the hypothesis that climbers rely not only on hold position but also on shape, size, and orientation when estimating difficulty. While LSTMs also benefited from the enriched input, the gains were less pronounced, likely due to their limited capacity to capture local spatial-semantic patterns.

B. *Model Behavior and Training Dynamics*

Training curves revealed rapid convergence and early overfitting, especially for models trained on spatial-only input. This pattern suggests that the architectures may be overparameterized for the input or that the representation lacks sufficient complexity to support extended learning. Attempts to mitigate overfitting included architectural simplification, the use of dropout layers, and batch normalization. While these strategies were partially successful in narrowing the training-validation gap, they also led to reduced overall performance—highlighting a trade-off between generalization and expressive power.

Moreover, the early stopping criterion—set to a patience of 20 epochs—proved suboptimal, allowing models to train well beyond their optimal point. This resulted in unnecessary computation and time, reinforcing the importance of tuning early stopping based on observed learning dynamics rather than fixed heuristics.

C. *Semantic Embedding Evaluation*

The embeddings were analyzed using t-SNE to project them into two dimensions and allow visual inspection of the latent space. The projection revealed meaningful visual structure, with clusters aligning closely to official MoonBoard hold sets. Subsequently, K-Means clustering with $K = 3$ was applied to the projected space, motivated by the known partition of MoonBoard holds into three sets. The resulting clusters exhibited high semantic coherence, grouping visually similar holds together and supporting the effectiveness of the embedding approach.

D. *Strengths and Contributions*

The main contribution of this work lies in the integration of visual semantics at the hold level into predictive models of climbing difficulty. This study is among the first to incorporate image-based hold representations in MoonBoard grade estimation. The proposed pipeline is modular and scalable—new hold sets can be embedded without retraining—and the learned embeddings are reusable for downstream tasks such as clustering, visualization, or route generation.

E. Weaknesses and Limitations

Despite promising results, the approach has several limitations:

- **Label noise**: MoonBoard grades are user-generated and may contain inconsistencies or subjective bias, impacting model accuracy.
- **Visual embeddings learned in isolation**: The autoencoder was trained independently from the main prediction task. While it captured hold set structure, embeddings could be optimized further using weak labels or multitask learning.
- **Model simplicity**: Basic CNN and LSTM architectures were used to isolate the effect of input enrichment. More advanced models (e.g., transformers or attention-based networks) may yield better performance.
- **Overfitting**: Models showed early overfitting. Although early stopping and regularization were applied, further strategies could be explored.
- **Class imbalance**: The dataset is heavily skewed toward mid-level grades (e.g., 6B+, 6C), biasing model predictions. Future work should apply techniques like stratified sampling, weighted loss functions, or data augmentation to address this issue.

IX. Conclusions and Future Work

This study presented a modular approach to MoonBoard difficulty prediction by enriching spatial route representations with semantic features extracted from hold images. Using embeddings generated by a convolutional autoencoder, the models gained access to visual cues such as shape, size, and surface texture—information that is often intuitively leveraged by climbers. The integration of these features improved prediction accuracy, particularly in convolutional architectures, and enabled a more interpretable analysis of the hold space.

Importantly, the semantic embeddings demonstrated utility beyond prediction. Clustering and dimensionality reduction techniques revealed that the learned representations captured meaningful visual groupings aligned with official MoonBoard hold sets. This supports the broader applicability of semantic modeling in climbing-related systems, including route generation, set visualization, and assistive tools for coaching or grading.

Future work could build on this foundation by incorporating more expressive architectures. Transformer-based models may improve the modeling of global context, while Graph Neural Networks (GNNs) offer a principled way to capture relationships between holds. Furthermore, replacing the fixed embeddings with a trainable component integrated into the full model may lead to better task-specific feature learning and improved overall performance. Addressing dataset-level challenges, such as grade imbalance and label noise, will also be critical for deploying robust and fair models in real-world applications.

In closing, this work highlights the value of combining spatial and visual information in modeling climbing difficulty, and lays the groundwork for future systems that more closely reflect how climbers perceive and assess the challenges presented by a route.

Code and Reproducibility

The complete implementation—including the training notebook for all models, as well as the data preprocessing and collection scripts—is available at: https://github.com/Rodrigueezj/Climbing-ML

Supplementary Material

A narrated poster presentation summarizing this work is available at:

https://youtu.be/c8z17Zo_q5Q

Acknowledgment

This work was carried out as part of a course project for academic purposes. The contributions and insights from prior research served as a foundation for this study and are gratefully acknowledged.

References

[1] D. Scarff, "Whole-History Rating for Rock Climbing," *arXiv preprint arXiv:2001.05388*, 2020. [Online]. Available: https://arxiv.org/abs/2001.05388

[2] GradeNet Project, "Human Performance in Visual Grade Prediction on MoonBoard Problems," Unpublished survey dataset, 2023.

[3] D. Petashvili and M. Rodda, "Board-to-Board: Evaluating MoonBoard Grade Prediction Generalization," *arXiv*, vol. abs/2311.12419, 2023. [Online]. Available: https://arxiv.org/abs/2311.12419

[4] S. Nam, J. Gwak, D. Cho, J. Park, and J. Kim, "ecurrent Neural Network for MoonBoard Climbing Route Classification and Generation" *arXiv*, vol. abs/2102.01788, 2021. [Online]. Available: https://arxiv.org/abs/2102.01788

[5] K. Garcia and J. Rodríguez, "Learn to Climb by Seeing: Climbing Grade Classification with Computer Vision," CS231n Course Report, Stanford University, 2024. [Online]. Available: https://cs231n.stanford.edu/2024/papers/learn-to-climb-by-seeing-climbing-grade-classification-with-comp.pdf