Group 45 Project

Deep Learning KU, WS 2025/26

Team Members		
Last name	First name	Matriculation Number
Jusic	Enis	12004830
Bajkusa	Mateo	11943093
Karic	Esma	11834983

January 21, 2026

1 Introduction

1.1 Problem Description

Image classification is the baseline task for this project, which utilizes Vision Transformers (ViT) to recognize and assign a label to an input image. This is achieved by training the model to detect and classify different visual patterns present in the images.

In this project, images are processed as a sequence of patches, on which a self-attention mechanism is applied in order to recognize relationships across all inputs. This allows the model to build a more global understanding of the image instead of relying only on local features. The aim of the project is to successfully classify the numbers displayed on the jerseys of football players

1.2 The Dataset

The dataset chosen for this project is the Caltech Football Numbers (CaltechFN) dataset, which contains a large number of images of football players annotated with their jersey numbers (see Figure 1). The images are noisy and often of low visual quality, meaning that the numbers are not always clearly readable. Due to the nature of the sport, jersey numbers are frequently twisted, stretched, partially occluded, or blurred, which makes the classification task more challenging.

The dataset is open source and split into training and test sets. Validation splits are constructed from the training set using group-based cross-validation. Each split consists of a folder containing full-color images of varying dimensions and a corresponding JSON annotation file. Each annotation includes a bounding box around a jersey number and its associated class label. For this project, each bounding box is treated as an individual sample by cropping the image accordingly. Bounding boxes with a width or height smaller than 8 pixels are discarded, as they do not contain sufficient visual information for reliable classification.[2]

Figure 1: Example picture from the dataset

2 Methods

Vision Transformers (ViT) are a deep learning architecture that applies the Transformer model to images. A standard ViT is made up of the following components: image patching and embedding, where the images are split into small patches of fixed size and mapped to feature vectors; a self-attention mechanism, where the model learns relationships between patches and captures global patterns across the image; and positional encoding, which are added to the patch features to retain spatial information and help the model understand where each patch came from.[1]

The implementation for this project follows the standard ViT design. An input image is split into patches of size PxP, where P is the patch size.

2.1 Data Preprocessing

Each cropped jersey number region is resized to a fixed resolution of 96×96 pixels. This is done before being sent to the model. It improves the model's robustness and prevents overfitting. Besides that we applied data augmentation during the training phase. This includes random color jittering (modifying brightness, contrast, and saturation by a factor of 0.2 and hue by 0.05) and random grayscale conversion with a probability of 0.05. Finally, images are normalized using the standard ImageNet mean (0.485, 0.456, 0.406) and standard deviation (0.229, 0.224, 0.225) to stabilize the gradients during training. Figure 2 shows example cropped jersey number images used as input to the model. The samples highlight the challenging nature of the dataset, including motion blur, partial occlusion, low contrast, and deformation of digits.

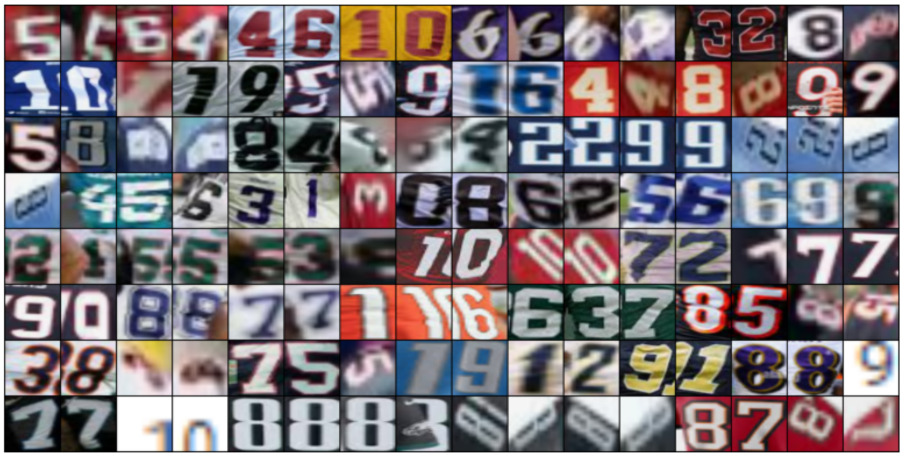

Figure 2: Example of a batch of cropped jersey number digits from the CaltechFN dataset. The samples illustrate noise, blur, occlusion, and distortion present in the data.

2.2 Patch Embedding

Instead of explicitly cutting patches and flattening them, patch embedding is implemented using a convolutional layer with kernel size and stride equal to the patch size. This produces one embedding per patch. After the convolution operation, the spatial dimensions are flattened and transposed, resulting in a sequence of patch tokens.

Each token is a learned $\mathcal{D}$-dimensional feature vector obtained by projecting the raw pixel values of one $\mathcal{P} \times \mathcal{P}$ patch, where $\mathcal{D}$ denotes the embedding dimension and $\mathcal{P}$ the patch size. The embedding dimension $\mathcal{D}$ and patch size $\mathcal{P}$ are treated as hyperparameters and vary across model configurations. This vector represents the visual information contained in the patch and serves as the basic unit processed by the Transformer.

2.3 Class Token and Positional Encoding

In order to perform classification, a learnable class token is introduced and appended to the patch token sequence. In addition, a learnable positional embedding is added to all tokens. This is necessary to preserve spatial information, as the self-attention mechanism itself is permutation-invariant and would otherwise ignore the relative positions of the patches within the image.

2.4 Multi-Head Self-Attention (MH-SA)

The input token sequence produced by the patch embedding layer is first mapped to queries, keys, and values using a single linear layer. The resulting tensor is then reshaped to separate the three components and the attention heads. Attention logits are computed using scaled dot products between queries and keys, followed by a softmax operation along the last dimension to obtain attention weights.

Dropout is applied to the attention weights, and the attention output is computed as a weighted sum of the value vectors. The outputs of all attention heads are then transposed and reshaped back to $R^{B \times N \times C}$, followed by a final linear projection and output dropout.

2.5 Feed-Forward Network (MLP)

Each token is processed independently by a two-layer feed-forward neural network with GELU activation. The hidden dimension of the MLP is set to four times the embedding dimension, as determined by the MLP ratio. Dropout is applied after each linear layer.

Each encoder block uses pre-layer normalization and residual connections to stabilize training and improve convergence.

2.6 Output Head

After the final Transformer encoder block, layer normalization is applied to the token sequence. Only the class token is retained and passed to a linear classification head, which maps the embedding to a vector of class logits corresponding to the jersey number classes.

2.7 Weight Initialization and Regularization

All learnable parameters are initialized following standard Vision Transformer practice. The positional embeddings, class token, and classification head weights are initialized using a truncated normal distribution, while the classification head bias is initialized to zero. Dropout is applied to positional embeddings, attention weights, projection layers, and MLP layers to reduce overfitting and improve generalization.

2.8 Loss Function

The model is trained using the categorical cross-entropy loss, which is commonly used for multi-class classification tasks. Let $\hat{y} \in R^K$ denote the output logits of the model for K classes. The softmax function converts logits into class probabilities,

$$p_k = \frac{\exp(\hat{y}_k)}{\sum_{j=1}^{K} \exp(\hat{y}_j)}.$$

Given the ground-truth class label t, the loss for a single sample is defined as

$$\mathcal{L} = -\log(p_t),$$

and the final loss is obtained by averaging over all samples in a batch.

This loss function is appropriate for the task, as the model outputs unnormalized class scores and the objective is to assign the highest probability to the correct jersey number.

3 Final Model Architecture

The final model is a Vision Transformer with a fixed input resolution of 96×96 pixels. Each image is split into non-overlapping patches of size 8×8, resulting in 144 patches per image. Each patch is converted into a 256-dimensional feature vector using a convolutional projection layer whose kernel size and stride match the patch size. The Transformer encoder consists of 6 identical blocks. Each block applies multi-head self-attention with 4 attention heads, followed by a feed-forward network with a hidden dimension four times the embedding size and GELU activation. Dropout with rate 0.1 is applied to attention weights and MLP layers for regularization. A learnable class token is prepended to the patch sequence and used as a global image representation. The final classification is produced by applying a linear layer to this class token.

4 Results

4.1 Training

The Vision Transformer model was trained using mini-batch gradient descent with the AdamW optimizer. Training was performed with a fixed batch size, and the categorical cross-entropy loss was minimized. Regularization was applied through dropout within the Transformer blocks as well as weight decay in the optimizer, where decay was applied only to weight parameters and excluded from bias terms, normalization layers, the class token, and positional embeddings.

To obtain a reliable estimate of model performance and reduce the risk of overfitting, a 5-fold cross-validation strategy was employed on the training data. The folds were constructed by grouping samples according to their image ID, ensuring that cropped jersey number regions originating from the same original image did not appear in both training and validation sets. This prevented data leakage and provided a more realistic evaluation of generalization performance.

During training, both training loss and validation loss were monitored, along with validation accuracy. Figure 3 shows the evolution of the validation loss over epochs. The loss decreases steadily for each architecture, indicating stable optimization and successful learning of the training data.

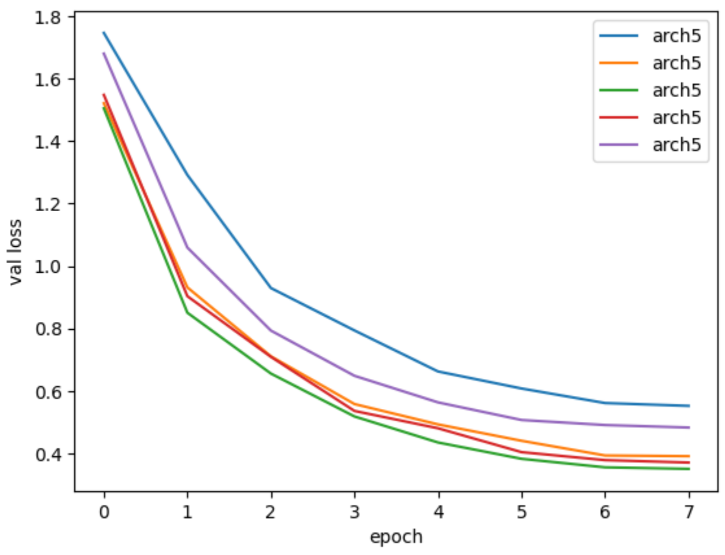

Figure 3: Validation loss across 8 epochs for the 5 selected architectures.

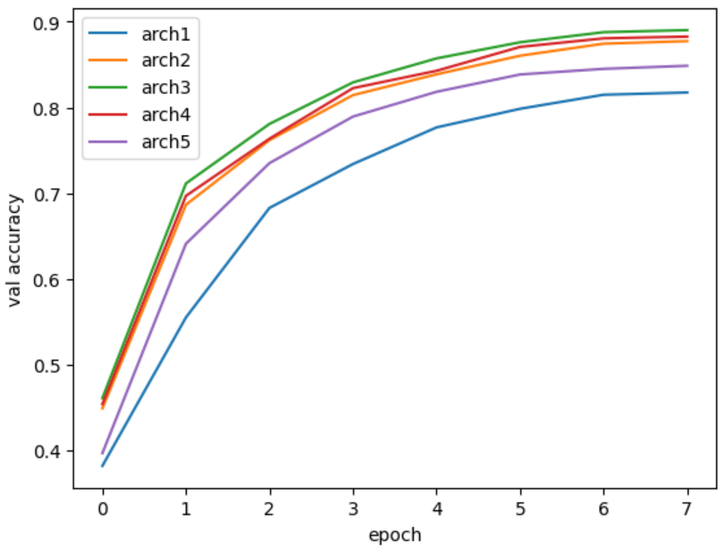

Figure 4: Validation accuracy across 8 epochs for the 5 selected architectures.

Figure 4 compares the validation accuracy curves of the different architectures and highlights the inferior performance of arch1.

4.2 Model Selection

To select a suitable model architecture, five Vision Transformer variants (arch1–arch5) were first evaluated using validation accuracy. arch1, utilizing large 12×12 patches, performed significantly worse (81.7% accuracy), proving that large patches lose too much detail.

Model	**Patch Size**	**Embed Dim**	**Depth**	**Heads**	**MLP**	**Dropout**
arch1	12	128	4	4	4.0	0.1
arch2	6	192	6	3	4.0	0.1
arch3	8	256	6	4	4.0	0.1
arch4	8	192	10	3	4.0	0.1
arch5	8	192	6	3	3.0	0.2

Table 1: Architectural hyperparameters explored during model selection.

Based on this initial comparison, arch3 (patch size 8, embedding dimension 256, depth 6) and arch4 (patch size 8, embedding dimension 192, depth 10) were selected as the most promising candidates.

Hyperparameter optimization was performed by evaluating several Vision Transformer architectures defined through a shared configuration structure. Each configuration specifies both architectural and training-related parameters, allowing for a reproducible comparison of different model variations.

The explored architectural hyperparameters include the patch size, embedding dimension, number of Transformer encoder blocks, number of attention heads, MLP ratio, and dropout rates. Five different architectures (arch1–arch5) were evaluated, each representing a different trade-off between model capacity and regularization. All models were trained using the same data preprocessing, loss function, optimizer, learning rate schedule, and number of epochs to ensure a fair comparison.

Model performance was evaluated using 5-fold cross-validation. Validation accuracy and validation loss were used as the primary criteria for model selection. The final architecture was chosen based on consistent validation performance across folds.

Arch3 achieved the highest mean validation accuracy of **89.58%**, slightly outperforming arch4 (89.03%) which is visuaized in Figure 5.

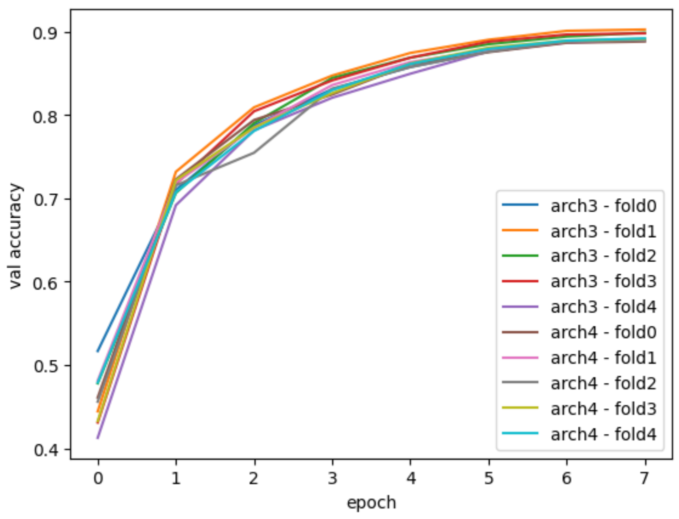

Figure 5: Validation accuracy across epochs for all five cross-validation folds of the selected architecture.

Figure 6 illustrates the validation loss for all five cross-validation folds. The loss decreases consistently and converges to comparable values across folds, indicating stable training and good generalization performance.

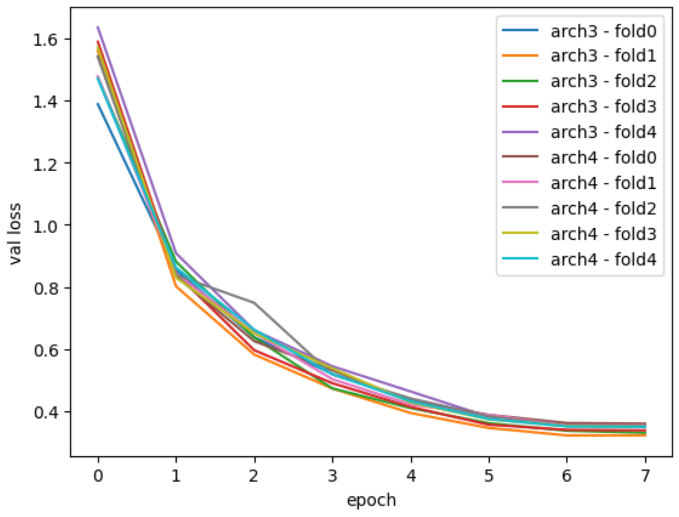

Figure 6: Validation loss across epochs for all five cross-validation folds of the selected architecture.

Model	**Mean Val Acc**	**Std Dev**
`arch3`	**89.58%**	0.58%
`arch4`	89.03%	0.14%

Table 2: Cross-validation results for the top architectures.

Based on these results, arch3 was selected as the final model, as it achieves the highest average validation accuracy as it can be seen in Figure 4.2 while maintaining stable performance across folds. Table 3 summarizes the final hyperparameter configuration selected based on this comparison.

Hyperparameter	**Value**
Input resolution	96×96
Patch size	8×8
Embedding dimension	256
Number of encoder blocks	6
Number of attention heads	4
η	$3e^{-4}$
min η	$1e^{-6}$
MLP ratio	4.0
Dropout rate	0.1
Optimizer	AdamW
Loss function	Cross-entropy
Batch size	128

Table 3: Final hyperparameter configuration of the selected Vision Transformer model.

4.3 Evaluation

After cross-validation, the trained model was evaluated on a held-out test set that was not used during training or validation. Model performance was measured using classification accuracy, defined as the proportion of correctly predicted jersey number classes. Accuracy is an appropriate evaluation metric for this task, as all classes are treated equally and the objective is to correctly identify the displayed jersey number.

The evaluation results show that the Vision Transformer is capable of learning meaningful visual representations from noisy input data and successfully classifying jersey numbers. The final model achieved a test accuracy of 90.34%, demonstrating strong generalization performance on unseen data.

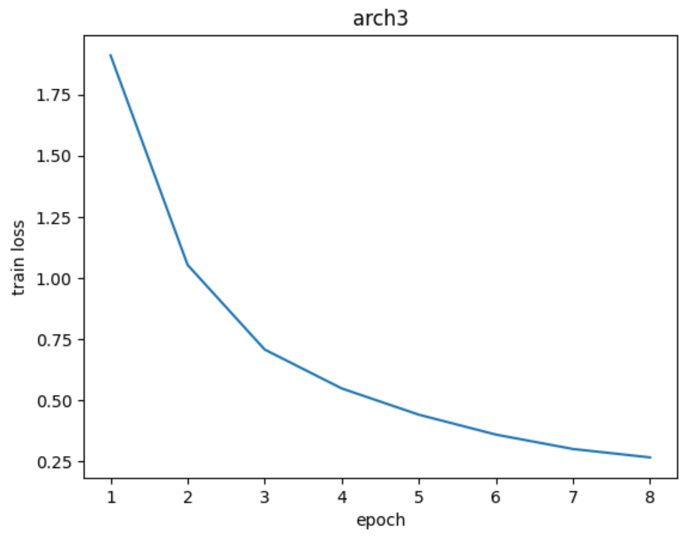

Figure 7: Final test loss across 8 epochs for the selected architecture.

5 Discussion and Conclusion

The primary challenge of the CaltechFN dataset is the high variance in image quality, motion blur, partial occlusions, and fabric distortions. Our results show that the Vision Transformer architecture is capable of learning meaningful patterns despite these noisy conditions of the dataset. The self-attention mechanism, which allows the model to capture global context across the entire image, likely contributed to this success by allowing the model to learn the identity of a digit even when parts of it were occluded or distorted.

Our experiments also highlight the importance of hyperparameter tuning when applying ViTs to smaller datasets. We observed that the model was highly sensitive to the balance between capacity and regularization. By employing a 5-fold cross-validation strategy, we were able to filter out unstable architectures and select a configuration (arch3) that generalized well to unseen data.

5.1 Conclusion

This project demonstrated the practical usage and results of using Vision Transformers for a image classification task of classifying football jersey numbers. We developed a custom training pipeline with data preprocessing, augmentation and a specific learning rate schedule to train a ViT from scratch. The final model achieved high classification accuracy, meaning it overcame the high variance in image quality. These results show that on a small to mid size dataset which is inherently complicated with all the variation in the data, the lightweight Vision Transformers are a promising tool.

References

[1] Erfan Khalaji. Vision transformers. https://medium.com/@erfankhalaji/vision-transformers-d0b1a2431dac, 2024. Medium blog post.

[2] Patrick Rim, Snigdha Saha, and Marcus Rim. Caltech football numbers (caltechfn), 2022.