MSc Artificial Intelligence
Master Thesis

Few-Shot Building Element Detection

via Self-Supervised Vision Transformer Patch Classification

by
T.J.G. Slik
15064898

July 3, 2025

36 EC
January 2025 - July 2025

UvA Supervisor:
Dr. A. Visser

Second reader:
Dr M.R. Oswald

Supervisor at TNO
Digital Built Environment:
MSc R.M.A Gueulet

Universiteit van Amsterdam

TNO innovation for life

Abstract

This thesis develops a computer vision methodology for detecting building elements in streetview imagery to support the Netherlands' climate initiative of renovating 1.5 million homes by 2030. The approach addresses the critical challenge of limited labeled training data in domain-specific building element detection by leveraging self-supervised Vision Transformers (ViTs) in a few-shot learning framework.

The methodology extracts patch-level embeddings from streetview images using pre-trained Vision Transformers, then trains lightweight classifiers to distinguish patches containing target building elements (solar panels, dormers, chimneys, roof windows, roof ventilation, parapets, and balconies) from background patches. Being effective with only 3-21 labeled examples per element, the approach achieves reliable detection performance despite severe data constraints.

Key technical contributions include: (1) demonstrating that general-purpose self-supervised representations (DINOv2) outperform specialized domain-specific retraining when paired with sophisticated classifiers; (2) developing an adaptive cropping strategy that improves patch selection by aligning bounding box annotations with ViT patch grids, increasing positive patch yield by up to 10x for small elements; and (3) introducing Random Forest Mixture of Experts (RF-MoE) for patch embedding classification, achieving superior performance over other classifiers.

Experimental evaluation on Dutch streetview imagery demonstrates the methodology's effectiveness, with the DINOv2 + RF-MoE configuration achieving 0.969 Average Accuracy and Precision (AAP) across all building elements. Per-element analysis reveals excellent performance for larger, visually distinct elements (perfect scores for dormers and roof windows) while highlighting challenges with smaller features.

The research validates the practical application of self-supervised Vision Transformers in specialized domains with limited labeled data, providing a scalable solution for automated building feature extraction that directly supports climate action initiatives through enhanced renovation planning capabilities.

Contents

Chapter 1 – Introduction

This thesis was written during an internship at TNO, contributing to a national initiative known as the "contingentenaanpak" [1] (Dutch for contingent approach or batch approach). This strategy aims to accelerate energy-efficient building renovations in the Netherlands by identifying homes suitable for standardized renovation solutions. Rather than inspecting individual buildings to assess applicability, the approach relies on collecting address level data about the Dutch housing stock and using artificial intelligence to automatically identify homes that match the criteria for energy-saving renovations.

TNO's *Clustertool* is the heart of this methodology, which is a classifier using over 50 building-related, address level features, such as energy performance, gas usage, facade composition, and demographics. These features allow the classifier to identify buildings that are suitable for specific renovation solutions. For buildings that qualify, owners will be approached with lowered-cost renovation offers, thanks to standardized solutions, and bulk acquirement of labor and materials. This approach directly supports the Netherlands' climate goals of renovating 1.5 million homes by 2030. The objective of this thesis is to expand the address-level dataset with detectable building elements extracted from streetview imagery.

The central contribution of this thesis is **a computer vision methodology to detect specific building elements in streetview images** from Google Maps or Cyclomedia to add to the per address features and enhance the Clustertool's predictive capabilities. Specifically, the method aims to detect the key elements: *solar panels*, *dormers*, *chimneys*, *roof windows*, *parapets*, and *balconies*, for which their presence will subsequently be encoded as boolean features in the Clustertool input.

However, implementing this computer vision methodology presents significant technical challenges. The primary obstacle is the scarcity of labeled training data for domain-specific building elements, which are rarely represented in conventional computer vision datasets. To address this limitation, the proposed approach leverages a self-supervised Vision Transformer (ViT) combined with a lightweight classifier in a few-shot learning framework.

The methodology employs a self-supervised ViT, pre-trained on large-scale unlabeled imagery, which has learned local and meaningful image representations in the form of patch embeddings. Using only 3 to 21 labeled examples per architectural element, a lightweight classifier is trained to distinguish whether a given patch embedding corresponds to the element of interest. This ViT-classifier combination can then process new, unlabeled streetview images by extracting patch embeddings and classifying them to detect the presence of target building elements. This approach achieves reliable detection performance despite the severe constraints on labeled training data, making it practical for real-world deployment where extensive manual annotation is prohibitively expensive.

[1] https://www.tno.nl/nl/newsroom/insights/2024/11/contingentenaanpak-bewijst-effectiviteit/

Chapter 2 – Background Research

This work proposes a method of using self-supervised vision transformers (ViTs), and performing classification on their patch embeddings. This approach will be implemented for detecting building elements in street-view images under a few-shot learning setting. The combination of self-supervised ViTs dense embeddings and few-shot detection explored in this thesis was inspired by several key research directions. In recent years, self-supervised ViTs have emerged as a foundation for significant advances across a wide range of computer vision tasks. Notably, Kaiming He, X. Chen, et al. (2021) and Caron et al. (2021) demonstrated that transformer-based self-supervised models can learn rich and generalizable visual representations from large-scale unlabeled data, rivaling or even surpassing supervised baselines in various downstream applications. These models have been successfully used in dense prediction tasks such as semantic segmentation and object detection, where patch-level representations enable fine-grained image understanding. Self-supervised ViTs can in some cases be applied directly without additional training, as demonstrated in LOST (Siméoni et al., 2021), or serve as a foundation integrated within complex pipelines such as Grounding DINO (Liu et al., 2024). This work focuses on developing a method that leverages self-supervised ViTs into an adaptable methodology to detect highly domain specific elements in a few-shot setting.

2.1 Emerging properties from Self-Supervised Vision Transformers

Self-supervised ViTs have become common technology in modern computer vision, providing versatile and powerful image representations. Even without any modifications, self-supervised ViT methods show themselves to be directly usable, as in segmentation.

DINO DIstillation with NO Labels (DINO) is a self-supervised learning framework for learning image representations (Caron et al., 2021) using a Vision Transformer (ViT) (Dosovitskiy et al., 2021). A ViT is trained through a student-teacher distillation setup where both networks share the same architectures but receive different input views. The training process begins with a multi-crop augmentation strategy that generates global crops (covering $> 50\%$ of the image) and local crops (covering $< 50\%$ of the image) from each input image. The teacher network only processes the global crops, while the student network receives all crops including both global and local views. During forward passes, both networks output feature representations from their [CLS] tokens, which are then passed through projection heads and normalized using a temperature-scaled softmax. The core learning objective is a cross-entropy loss computed between the student's outputs and the teacher's outputs, excluding identical crop pairs. The student network is optimized through backpropagation and gradient descent, while the teacher network parameters are updated using an exponential moving average of the student's weights. This setup forces the student to learn local-to-global correspondences, enabling the ViT to develop robust visual features without requiring labeled data. As a result, the model learns to

produce similar representations not only for different views of the same image, but also for semantically or visually similar images, effectively grouping them together in the learned feature space based on shared visual characteristics.

Remarkably, Caron et al. (2021) demonstrate that through this training setup, the self-attention maps from the final [CLS] token naturally highlight important regions which is visualized in Figure 2.1, effectively segmenting primary objects in object-centric datasets without any supervision. This emergent behavior is a first step toward automatic segmentation, enabling the model to isolate meaningful structures in the image without any supervision.

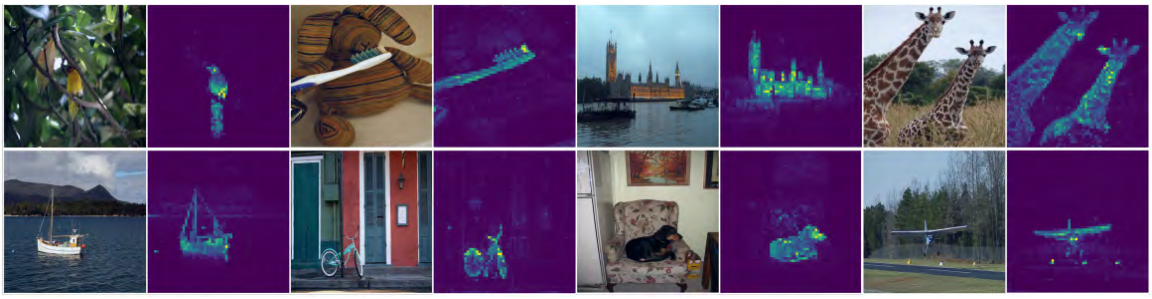


Figure 2.1: Caron et al. (2021) figure 1, displaying [CLS] token self-attention maps.

DINOv2 Oquab et al. (2023) introduces DINOv2, which applies the same self-distillation principles as the original DINO, but is trained on a much larger and more diverse dataset (LVD-142M) with improved and more robust training routines. As a result, DINOv2 produces more general and transferable visual representations, achieving state-of-the-art performance across a wide range of downstream tasks.

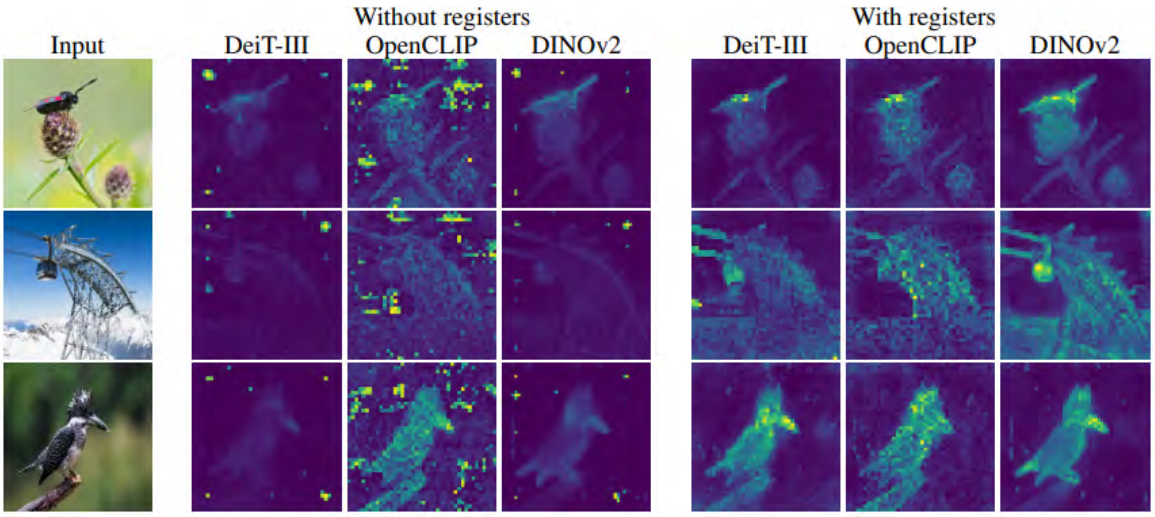


Figure 2.2: Darcet et al. (2024) figure 1, displaying [CLS] token attention maps for large scale vision transformers with and without registers.

DINOv2 with registers Darcet et al. (2024) discovered that scaling up vision transformers leads to artifacts in feature maps, where high-norm tokens appear primarily in low-informative background areas of images. These artifacts are illustrated in Figure 2.2 which, on the left side, shows attention maps from vision transformers without register tokens. The high-norm background tokens correspond to tokens that are repeatedly used for internal computations,

resulting in noisy and spatially unstable attention maps. This behavior limits the model's ability to form coherent and smooth representations, particularly for downstream tasks. The authors argue that this phenomenon arises because of a lack of space to process global information, causing it to accumulate in semantically meaningless background tokens. To mitigate this issue, they introduce *register tokens*, learned tokens similar to the [CLS] token, which act as explicit storage units for global information. These register tokens absorb the high-norm activations that would otherwise contaminate background regions, leading to cleaner and more interpretable attention maps, as shown in the right-hand side of Figure 2.2. This modification leads to improved performance, particularly on dense visual prediction tasks that depend on consistent dense ViT outputs, which Darcet et al. (2024) demonstrates on the aforementioned method called LOST (Siméoni et al., 2021).

2.2 Taming self-supervised vision transformers

Although self-supervised vision transformers, particularly in DINO models, demonstrate promising capabilities, their generated masks tend to be coarse and noisy, primarily designed for object-centric datasets, and are not readily applicable in practical scenarios. Research efforts such as LOST (Siméoni et al., 2021), MOST (Rambhatla et al., 2023), STEGO (Hamilton et al., 2022), and U2Seg (Niu et al., 2024) have aimed to "tame" these ViTs to produce finer-grained predictions without the need for retraining. A shared characteristic of these methods is their reliance on dense patch-level representations extracted from the final layers of ViTs, which are subsequently refined using clustering algorithms and patch embedding similarity measures.

Object discovery Siméoni et al. (2021) and Rambhatla et al. (2023) demonstrate that by selecting patches whose key vectors (from the transformer's attention mechanism) are maximally dissimilar from the rest, one can reliably identify patches corresponding to foreground objects, essentially segmenting out foreground objects as instances. The underlying assumption is that object regions manifest as distinct modes in the attention feature space.

While their use of dense ViT features without any retraining is inspiring, this thesis focuses on detecting elements based on semantic understanding of their visual properties. In this regard, the approaches LOST and MOST fall short, as they are class-agnostic, making them unsuitable for tasks involving semantics. This limitation motivates the exploration of methods that can move beyond generic object discovery toward interpretable and semantically structured segmentation outputs.

Fully Self-Supervised Segmentation The approaches of STEGO (Hamilton et al., 2022) and U2Seg (Niu et al., 2024) demonstrate promising results for fully self-supervised semantic segmentation. Both methods utilize clustering techniques applied to ViT features, subsequently aligning the resulting pseudo-labels with ground truth classes in the training dataset.

STEGO produces final segmentation masks through a multi-stage process: it first extracts dense patch-level features from a DINO-pretrained ViT, then applies a segmentation head to project these features into a lower dimension, and finally employs contrastive learning to form compact clusters that correspond to semantic regions. The method uses feature correspondence tensors to establish pixel-level relationships across image crops, enabling the clustering algorithm to group semantically similar pixels into coherent segments.

U2Seg takes a more comprehensive approach by generating masks for multiple segmentation tasks simultaneously. It first creates high-quality discrete semantic labels by clustering instance masks obtained from MaskCut and DINO features, then combines these "things" pixels with "stuff" pixels generated by STEGO to produce comprehensive pseudo semantic labels for every pixel in the image. The final universal segmentation model, a ResNet50 based model (Kaiming He, Zhang, et al., 2015), is trained on these pseudo-labels, resulting in a unified framework capable of producing instance, semantic, and panoptic segmentation masks from a single inference pass. The final step to create a practically useful model involves matching the masks produced by the segmentation models based on pseudo-labels to ground truth labels, which both STEGO and U2Seg accomplish through Hungarian matching (Kuhn, 1955).

While these achievements are impressive in their ability to produce detailed segmentation masks without human supervision, for this research the clustering approaches are particularly interesting for accurately detecting the presence of architectural elements, where detailed perfectly outlined segmentation masks are not necessary. The key insight from these works is that patches displaying similar visual content exhibit strong correspondence in the feature space, which is visualized in Figure 2.3. This supports the hypothesis that dense patch representations inherently encode semantic information about the elements present within each patch. This leads to the idea that dense patch representations from ViTs may also be directly utilized to identify elements of interest within images, without requiring the complex segmentation pipelines created by STEGO and U2Seg.

Figure 2.3: Figure 2 from Hamilton et al. (2022) showing DINO patch correspondence of regions that correspond to the blue, red, and green cross.

Retraining ViTs to Enhance Patch Embedding Quality Rather than externally modifying a ViT's behavior, NeCo (Pariza et al., 2025) introduces a dense post-pretraining approach that refines patch-level representations through continued self-supervised learning. As illustrated in Figure 2.4, NeCo employs a teacher-student framework where both models process different augmented views of the same image, with the teacher model using exponential moving averages of the student's parameters to ensure stable learning dynamics.

NeCo operates through a systematic four-step process:

1. **Augmented Views and Encoding:** A pretrained Vision Transformer (e.g., DINOv2 with 4 register tokens) is used to encode augmented views of input images. These views are processed by both student and teacher models.

2. **Feature Extraction and Neighbor Computation:** Dense patch-level features are extracted from each model. For every patch, nearest neighbor relationships are computed with respect to reference patches sampled from other images in the batch.

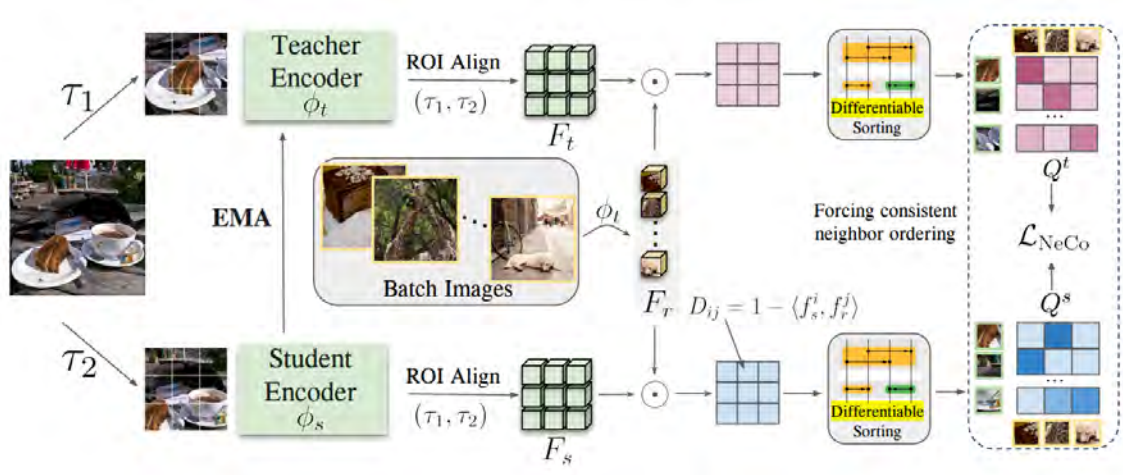


Figure 2.4: Figure 1 from Pariza et al. (2025) showing the training pipeline of NeCo. An input image is augmented into two views, processed by student and teacher encoders. Features are aligned and compared to reference features from other images. Pairwise distances are computed and sorted to enforce nearest neighbor consistency across views using the NeCo loss.

3. **Differentiable Sorting:** The patch-wise distances are sorted using a differentiable sorting mechanism to obtain ordered lists that preserve fine-grained spatial similarity information.

4. **Patch Neighbor Consistency Loss:** The core objective, Patch Neighbor Consistency Loss ($\mathcal{L}_{\text{NeCo}}$), enforces consistency between the student and teacher ordered neighbor lists, promoting robust patch-level representations invariant to augmentation.

The core innovation lies in how NeCo computes and enforces consistency in patch-level nearest neighbor relationships. Given augmented views V_1 and V_2 of an input image, the student and teacher models ϕ_s and ϕ_t extract spatially aligned dense features $F_s \in \mathbb{R}^{N' \times d}, \quad F_t \in \mathbb{R}^{N' \times d}$ respectively.

For each patch feature, distances are computed using cosine similarity relative to reference patches $F_r \in \mathbb{R}^{R \times d}$ sampled from the batch:

$$D_s(i,j) = 1 - \frac{\langle F_s^i, F_r^j \rangle}{\|F_s^i\| \, \|F_r^j\|}, \quad D_t(i,j) = 1 - \frac{\langle F_t^i, F_r^j \rangle}{\|F_t^i\| \, \|F_r^j\|}$$

for $i = 1, \ldots, N'$ and $j = 1, \ldots, R$.

Rather than using hard sorting (which is non-differentiable), NeCo applies a *differentiable sorting algorithm* to convert each row of the distance matrices D_s and D_t into *soft permutation matrices*:

$$Q_s^i, Q_t^i \in \mathbb{R}^{R \times K}$$

Each (r, k)-th entry of Q_s^i (or Q_t^i) denotes the probability that reference patch r is the k-th nearest neighbor to patch i.

The loss for enforcing bidirectional consistency between the sorted neighbors ensures hat the teacher's view of which reference patches are most similar (i.e., ordering in Q^t) is mirrored by the student's ordering Q^s, and both $Q^t \to Q^s$ and $Q^s \to Q^t$ are penalized if inconsistent. The

loss is defined as:

$$\mathcal{L}_{\text{NeCo}} = \sum_{i=1}^{N'} \left[\mathcal{L}_{\text{CE}}(Q_i^t, Q_i^s) + \mathcal{L}_{\text{CE}}(Q_i^s, Q_i^t) \right]$$

where the cross-entropy loss $\mathcal{L}_{\text{CE}}$ is given by:

$$\mathcal{L}_{\text{CE}}(Q_i^t, Q_i^s) = -\sum_{j,k} Q_i^t(j,k) \log Q_i^s(j,k)$$

This formulation ensures that nearest-neighbor rankings remain consistent across both teacher and student models over different views, providing richer supervision compared to binary contrastive learning. The differentiable sorting mechanism enables gradient flow through the permutation process, facilitating learning of nuanced spatial relations while maintaining the semantic ordering of patch similarities.

NeCo's post-pretraining process produces a refined feature space where patches representing the same semantic content exhibit similar embeddings, while patches from different semantic categories demonstrate distinct embeddings. This enhanced patch-level discrimination proves particularly valuable for tasks requiring fine-grained spatial understanding, as it improves the model's ability to identify and distinguish between different visual elements within complex scenes. The approach effectively bridges the gap between general-purpose pretrained features and task-specific requirements for precise patch-level understanding, establishing a promising foundation for accurate patch-level classification in domain-specific element detection tasks.

2.3 Patch Embedding Classification

Research has demonstrated that patch-level classification can be highly effective for few-shot learning scenarios. Recent work by Jiang, Cui, and Kun He (2024) proposes selecting class-relevant patch embeddings by calculating similarity between class embeddings and patch embeddings, retaining only the top-ranked patches to form comprehensive image representations. Similarly, Hao et al. (2023) introduce class-aware patch embedding adaptation that makes patch embeddings class-relevant through constant interaction with class-aware embeddings, demonstrating significant improvements on benchmark datasets. These research methods both employ MLPs in their patch embedding-based image classification approaches, with theoretical foundations showing that patch-level routing in mixture-of-experts can provably reduce sample complexity by filtering label-irrelevant patches.

While their methodology is not specifically designed for few-shot settings, Pariza et al. (2025) utilize k-means clustering and an MLP together with Hungarian matching (Kuhn, 1955) to bridge the gap between their meaningful patch embeddings and assigning class labels to ViT patches. The Hungarian algorithm serves as a critical component for resolving the label assignment problem inherent in clustering approaches, where cluster identifiers do not inherently correspond to semantic class labels.

These MLP-based approaches provide a solid baseline for comparison in patch embedding classification, establishing the effectiveness of neural network classifiers for patch-level feature discrimination. However, our investigation extends beyond traditional neural network classifiers to explore more sophisticated methods including ensemble techniques like Random Forest Mixture of Experts (RF-MoE) and support vector machines, which may offer enhanced performance in few-shot learning scenarios by leveraging different classification paradigms and ensemble strategies.

Support Vector Machine (SVM) with Radial Basis Function (RBF) kernel Support Vector Machines with Radial Basis Function (RBF) kernels provide a systematic approach for classifying high-dimensional patch embeddings. The theoretical foundation for SVMs is comprehensively covered in Bishop's "Pattern Recognition and Machine Learning" (Bishop, 2006).

Kernel Transformation and Feature Mapping. The RBF kernel transforms the input space into a higher-dimensional feature space using:

$$k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

where γ controls the kernel width. This kernel function implicitly maps the input vectors into an infinite-dimensional feature space through the "kernel trick," allowing the SVM to operate in this high-dimensional space without explicitly computing the feature mapping $\varphi(x)$ (Bishop, 2006).

The kernel represents the inner product $k(x, x') = \varphi(x)^T \varphi(x')$ in the transformed space, enabling non-linear decision boundaries in the original input space.

Optimization Problem with Regularization. The SVM finds the optimal separating hyperplane by solving the constrained optimization problem:

$$\min_{w,b,\xi} \quad \&\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

subject to $\quad y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \;\&\; \xi_i \geq 0, \quad i = 1, \ldots, n$, where C is the regularization parameter controlling the trade-off between margin maximization and training error minimization (Bishop, 2006). The slack variables ξ_i allow for misclassified points, making the SVM robust to outliers.

Dual Formulation and Support Vectors. Using Lagrange multipliers, the optimization problem transforms into its dual form, where the solution depends only on the support vectors (training points with non-zero Lagrange multipliers α_i). The decision function becomes:

$$f(x) = \text{sign}\left(\sum_{i=1}^{n}\alpha_i y_i\, k(x_i, x) + b\right)$$

where only support vectors contribute to the classification decision (Bishop, 2006).

This approach leverages the kernel trick to handle non-linearly separable data while maintaining computational efficiency through the sparse representation using support vectors. The RBF kernel's ability to create smooth decision boundaries makes it particularly interesting candidate for the classification of distinguishing between patches of different building elements.

Random Forest Mixture of Experts (RF-MoE) The Random Forest Mixture of Experts (RF-MoE) represents a hybrid architecture that combines the ensemble strength of random forests with the specialization capabilities of mixture of experts frameworks. As illustrated in Figure 2.5, this approach employs multiple random forest experts coordinated through a clustering based gating mechanism that routes patch embeddings to the most appropriate expert based on their feature vectors.

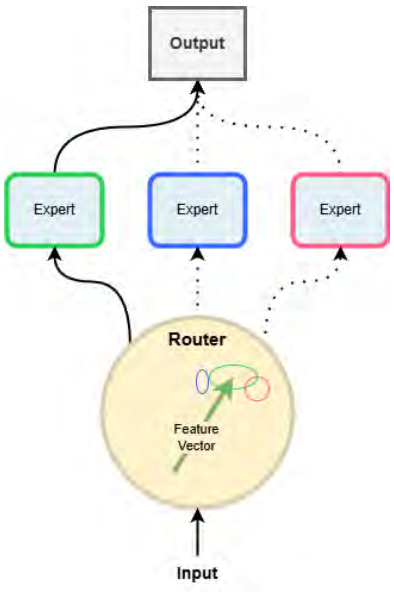


Figure 2.5: Schematic illustration of a Mixture of Experts architecture with clustering-based routing. The feature space is partitioned into distinct clusters, each assigned to a specialized expert responsible for predictions within that region. In this visualization, a feature vector located within the green cluster is routed to the corresponding green expert for classification.

The theoretical foundation for expert specialization is well-established in recent research. For example, Nielsen et al. (2025) demonstrate that sparse mixture of experts covering semantically distinct areas of the feature space achieve both fast convergence and true expert specialization within their respective regions. This principle is particularly relevant for patch embedding classification, where patches containing similar architectural elements may cluster together in the feature space, while patches representing different element types (e.g., balconies versus chimneys) are expected to naturally separate into distinct regions. The MoE framework demonstrates robustness across different granularities of class separation in the feature space, enabling the deployment of specialized experts to handle distinct regions and leading to more focused and effective classification models.

Random Forest classifiers have proven their effectiveness when working with Vision Transformer representations. Research on ViT CLS token classification shows that Random Forest classifiers achieve competitive performance alongside SVMs (Strano Moraes et al., 2025). This effectiveness naturally extends to patch-level classification, where Random Forest's inherent ability to handle high-dimensional feature vectors and provide robust ensemble predictions makes it ideally suited for processing ViT patch embeddings.

The RF-MoE architecture specifically addresses critical challenges in few-shot learning scenarios through three key mechanisms. First, the gating network enables routing of patch embeddings based on their presence in the feature space, ensuring each expert can be trained toward their specialized feature space region. Second, the Random Forest component provides inherent robustness against overfitting (Breiman, 2001), a crucial advantage when working with limited training examples typical in few-shot learning contexts. Finally, the dual ensemble nature of both the MoE framework and individual Random Forest experts creates multiple levels of prediction aggregation, potentially improving generalization performance across diverse patch embeddings from building elements occupying different semantic regions of the feature space.

While there is limited explicit research specifically combining mixture of experts architectures

with clustering-based gating mechanisms and random forest classifiers, the theoretical foundations and practical intuitions discussed earlier suggest that this hybrid approach holds significant promise. Given the strengths of random forests in handling high-dimensional data and the specialization capabilities of mixture of experts frameworks, it is an interesting and valuable direction to empirically investigate how such an RF-MoE model performs, particularly in scenarios involving patch embedding classification.

Chapter 3 – Methodology

The overall approach in this project is best described as the detection of building elements in a few-shot setting with self-supervised vision transformers and the classification of its patch embeddings.

The methodology of this project, as visualized in Figure 3.1, follows the following steps process:

- Annotated images with a certain element of interest (a dormer in Figure 3.1) are fed through a vision transformer.
- When cutting image in a grid style to produce patches, each patch with element of interest in it based on the annotation is selected as positive, others are negative.
- The ViT processes the patches and produces meaningful embeddings for each patch.
- For each architectural element, the ViT is run on 3 to 21 annotated images, and all positive and negative patch embeddings are collected.
- A classifier is then trained in the patch embedding space to distinguish between patches containing the element of interest and those that do not.
- The same ViT and a now trained classifier are subsequently applied to new, unseen images to identify patches likely to contain the target building element.

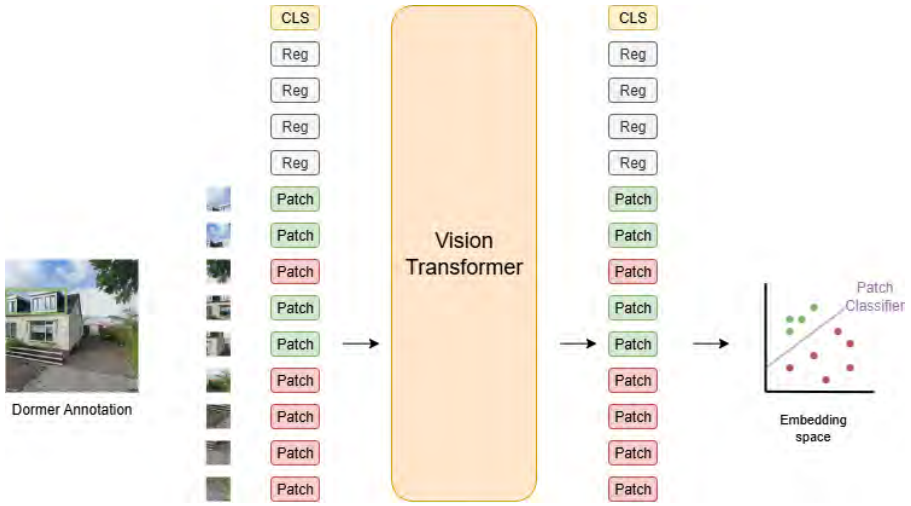
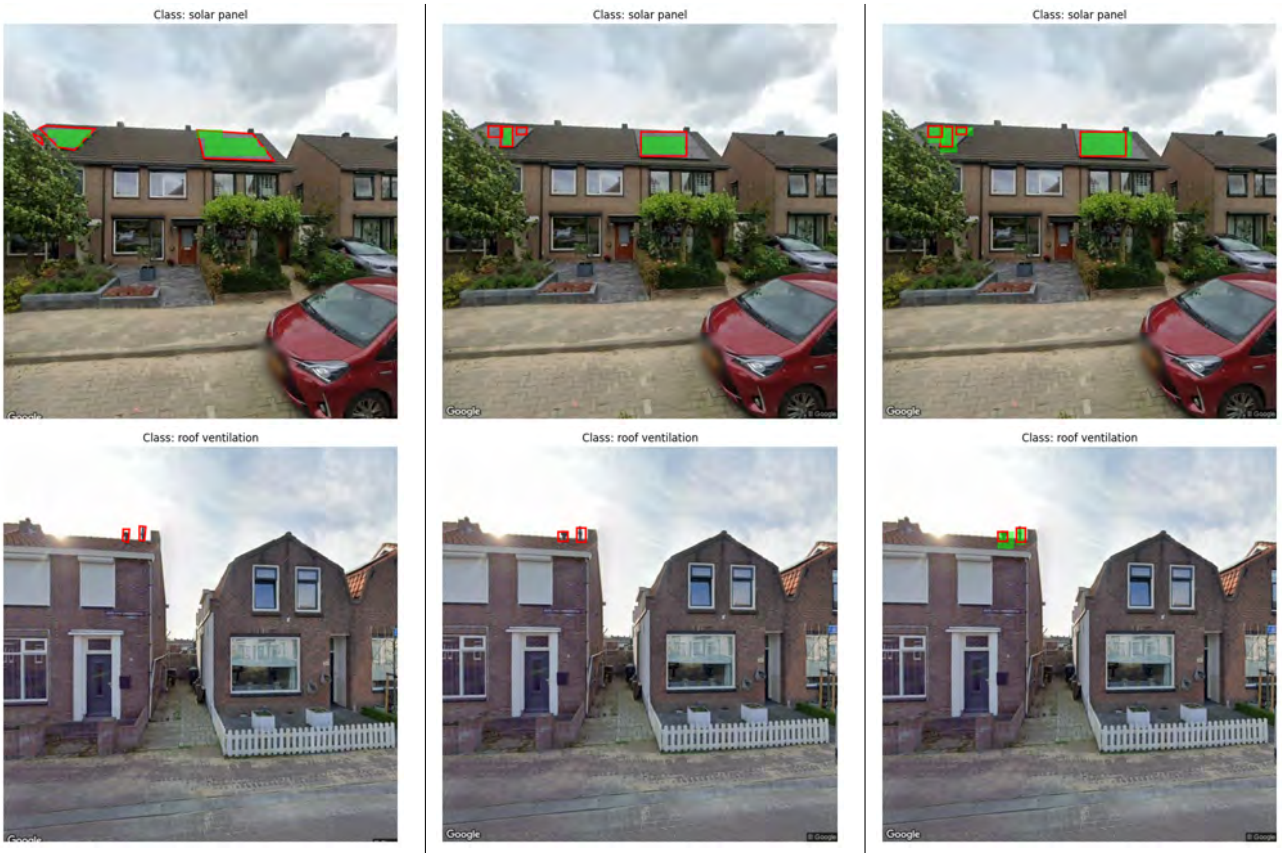


Figure 3.1: Schematic overview of the proposed pipeline using a dormer annotation as an example. From left to right: The annotated image is divided into 14×14 pixel patches. Patches containing the element of interest are labeled as positive samples (green), while those without are labeled as negative (red). The ViT outputs a meaningful embedding for each patch. A classifier is then trained in this embedding space to distinguish between positive and negative patches.

This pipeline enables robust detection of specific building elements with minimal labeled data, leveraging the representational power of self-supervised ViTs and the efficiency of lightweight patch embedding classifiers in a few-shot setting.

3.1 Patch Selection

The goal of this project is to enrich address-level data for a defined set of addresses in the Netherlands by extracting features from street-view imagery. A team of experts from the Building Energy Systems department at TNO has created a ranked list indicating the expected impact of having data on each specific building feature. Based on this list, a selection was made of features considered relevant for this project, specifically, those with a detectable presence in imagery and expressible as a boolean indicating whether or not the feature is present on the building. The building elements identified as important for detection are defined as: "*solar panel*", "*dormer*", "*chimney*", "*roof window*", "*roof ventilation*", "*parapet*", and "*balcony*". For each of these elements of interest, up to 21 street view images were to be annotated. The annotation process happened manually and finding the right strategy is essential.



(a) Polygon t=0.7 annotation

(b) Bounding box annotation

(c) Bounding box annotation with adaptive cropping

Figure 3.2: A visualization of different annotation styles influencing how patches are selected for their embeddings to be included in positive samples for (in this case) the elements of solar panel and roof ventilation.

3.2 Annotation and Patch Selection

Obtaining a sufficient number of patch embeddings for the elements of interest is highly desirable for the effectiveness of this few-shot detection approach. The well-known machine learning principle that "more data leads to better performance" certainly holds true in this context. However, beyond quantity, the quality of annotations is equally important. The quality of annotation can be described by whether an annotation truly contains an element of interest. Both the annotation process and the strategy used for selecting patches have a strong influence on the number of patches that are collected, and the areas in the image covered by the collected patches. In this section the development of the annotation and patch selection strategy is discussed.

An element of interest in an image can be annotated with a **polygon** or a **bounding box**. A polygon can more accurately outline an element of interest, but takes a long time to annotate with. A bounding box is faster to annotate with, but may be a rough outline of an object. Figure 3.2a and Figure 3.2b show manually annotated images in polygon and bounding box style respectively. In the polygon style annotation, patches are selected when they have an threshold (t=0.7), a minimum intersection over union threshold. For the bounding boxes, only patches that fall within the bounding boxes are selected as positives. The positive patches are highlighted in green.

Adaptive cropping Performing classification on patch embeddings with only a few annotations limits the available training data for the patch embedding classifier. It is therefore crucial to maximize the number of patch embeddings extracted from each image that accurately cover areas of interest, ensuring that enough patch embeddings per element of interest are collected.

In many cases, the image and its associated bounding boxes are poorly aligned. A bounding box may be too small to contain even a single 14×14 pixel patch, or it may be misaligned with the fixed 14×14 patch grid. As a result, many relevant patches are excluded from the set of positive example embeddings. Conversely, when patches are included based on an area-overlap threshold, they may extend beyond the object of interest and capture irrelevant regions. These issues result in both undersampling and oversampling of the patch embeddings corresponding to the target object.

Figure 3.3 illustrates a proposed adaptive cropping strategy for aligning bounding boxes with the grid of ViT patches. In the standard patch grid overlay, the grid lines of the patch boundaries of the ViT do not align properly with the bounding boxes containing the elements of interest. As a result, patches that only partially cover the object would either be discarded or included. Discarding comes at the cost of including valuable training data for the patch classifiers, and including patches partially covering the object of interest comes at the cost of capturing excess background. The aligned grid overlay shows the result of the adaptive cropping approach. For every bounding box, the image is cropped to be realigned with the ViTs grid of 14x14 pixel patches from the top-left corner. It is also ensured that at least one patch is selected from a bounding box, also if it is smaller than the bounding box. Adaptive cropping ensures the number of relevant patches is maximal, and it reduces the inclusion of irrelevant surrounding area.

As shown in Table 3.1, annotation style and patch selection strategy strongly affect the number and diversity of positive patch embeddings for each building element. Polygon (with t=0.7) annotations work well for large elements like balconies, but yield very few positives for small

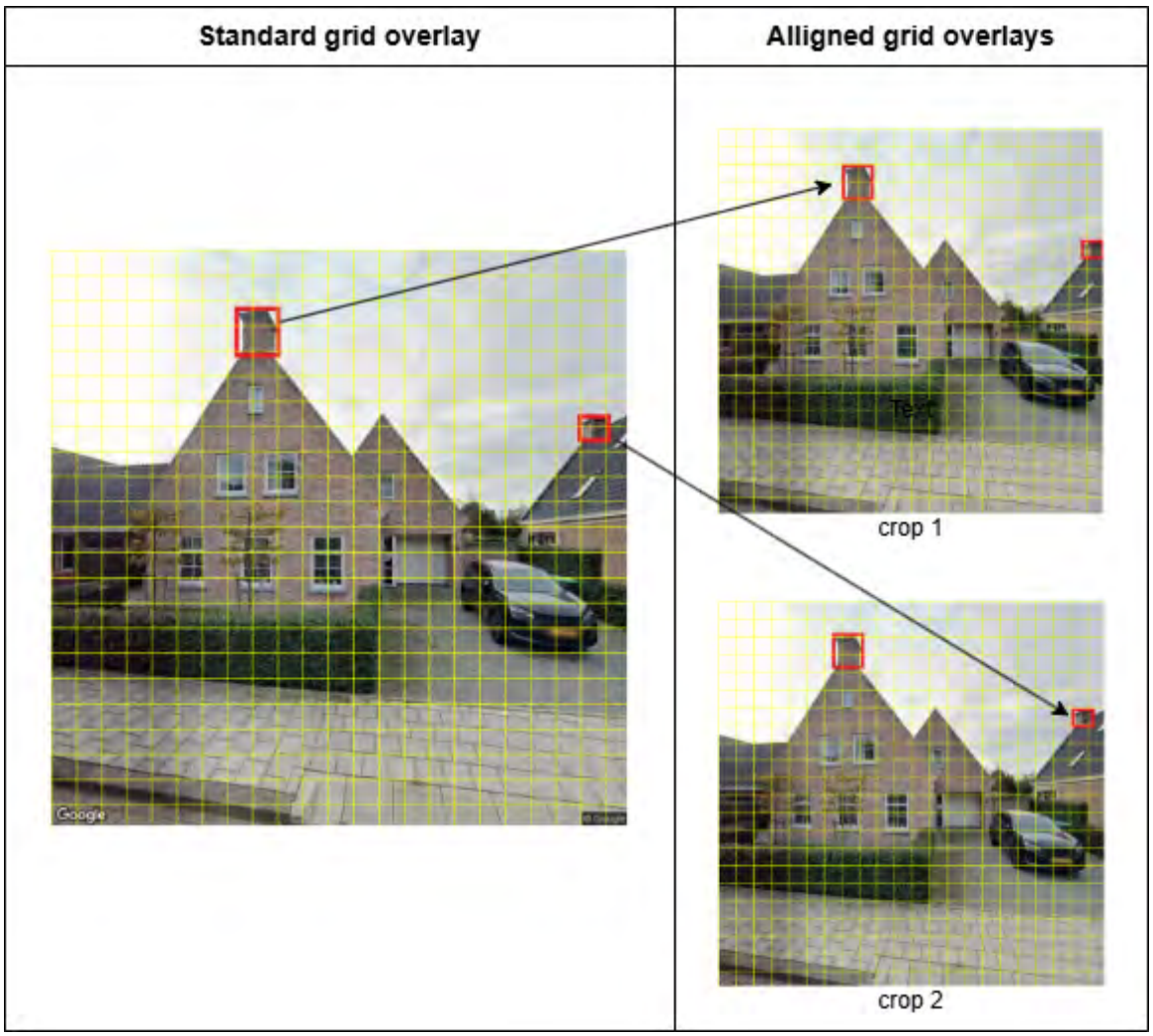


Figure 3.3: Visualization of the adaptive cropping strategy. In the standard grid overlay, the ViT patch grid does not align well with the bounding boxes of elements of interest. For each bounding box the images are cropped to improving coverage and alignment.

elements such as chimneys or roof ventilation.

Table 3.1: Count and ratio of positive patch embeddings per class across different annotation styles. "Count" represents the number of positive patch embeddings obtained, while "Ratio" indicates the proportion of positive embeddings relative to the total number of sampled patches.

Class	**Polygon**		**Polygon** $t = 0.7$		**Standard BB**		**BB Adaptive Crop**	
	Count	*Ratio*	*Count*	*Ratio*	*Count*	*Ratio*	*Count*	*Ratio*
Solar Panel	145	0.0073	269	0.0135	103	0.0066	244	0.0132
Dormer	148	0.0074	244	0.0122	234	0.0148	539	0.0286
Chimney	9	0.0004	50	0.0025	0	0.0000	57	0.0029
Roof Window	8	0.0004	37	0.0018	2	0.0001	69	0.0036
Roof Ventilation	1	0.0000	7	0.0003	8	0.0005	75	0.0039
Parapet	170	0.0086	282	0.0142	107	0.0068	530	0.0275
Balcony	827	0.0431	1107	0.0569	368	0.0233	867	0.0453

Bounding box annotations, especially when combined with adaptive cropping, greatly improve representation for smaller elements. Adaptive cropping ensures at least one patch is extracted per annotation, even for small or misaligned elements, leading to a substantial increase in posi-

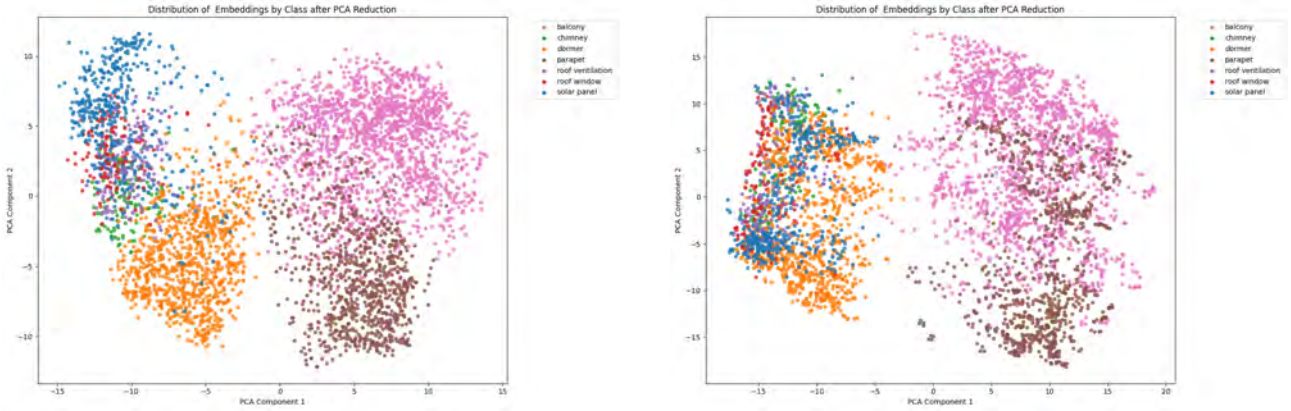
tive samples for challenging classes. For example, roof ventilation positives rise from 1 (polygon) and 8 (standard box) to 75 with adaptive cropping; chimneys increase from 9 (polygon) and 0 (standard box) to 57.

This is crucial in few-shot learning, where annotated data is limited. Adaptive cropping maximizes relevant training patches, particularly for less prominent elements, and reduces irrelevant background by aligning the patch grid with the object. Although adaptive bounding box cropping does not always yield the highest total number of positive samples, it consistently produces more positives for classes with smaller annotations because it ensures each annotation contributes at least one patch, improving representation for less prominent object classes.

In summary, adaptive cropping is highly effective for patch selection in this few-shot framework, ensuring sufficient and representative patch embedding training data and directly supporting robust classifier training.

3.3 Retraining with NeCo

Patch embeddings should meaningfully represent the visual content of each patch and ensure that patches depicting similar elements or spatial regions are close in the embedding space. To improve this property, the NeCo post-pretraining methodology (Pariza et al., 2025) was applied on a DINOv2 ViT base model with 4 register tokens and 768-dimensional embeddings. The model was trained for 50 epochs on a dataset of approximately 20,000 streetview images.



(a) Patch embeddings (PCA reduced) before retraining.

(b) Patch embeddings (PCA reduced) after retraining.

Figure 3.4: PCA visualization of patch embeddings before and after ViT retraining using NeCo for 50 epochs on 20,000 streetview images. The left subfigure shows embeddings prior to NeCo retraining, while the right subfigure shows embeddings post-retraining. All 7 elements of interest are visualized as points with a different color for each.

To assess the effect of NeCo retraining, the patch embeddings are visualized for each building element using PCA (Pearson, 1901), as shown in Figure 3.4. The left panel displays patch embeddings from the original DINOv2 model, while the right panel shows embeddings after NeCo retraining. Before retraining, certain classes—such as "dormer" and "solar panel"—are already well separated in the embedding space. After NeCo training, we observe that classes which typically appear in similar spatial regions (e.g., "dormer" and "solar panel") are pulled closer together, resulting in more overlap between their embeddings. In contrast, elements that tend to occur in different regions of the image (such as "dormer" and "balcony") seem to be

pushed further apart.

This behavior demonstrates that the NeCo methodology effectively encourages spatially similar elements to have more similar patch embeddings, enhancing spatial consistency in the learned feature space. However, this also introduces a potential trade-off: while NeCo increases the proximity of spatially related classes, it may also make it more challenging to distinguish between visually or spatially similar elements during downstream classification, as their embeddings become less separable.

3.4 Patch Classification

In this work, patch classification refers to the process of assigning a binary label to each patch embedding extracted from a Vision Transformer, indicating whether it contains the building element of interest or not. Given that each streetview image is divided into a large number of patches, it is essential that the chosen classifiers are lightweight and capable of rapid inference to efficiently process many patches per image.

A variety of lightweight classifiers are considered, each with distinct characteristics that may be advantageous depending on the structure of the patch embedding space. These include clustering-based methods, neural network models, kernel-based approaches, and ensemble techniques. The classifiers are used in a binary setting: for each element of interest, a separate classifier is trained to distinguish between positive patches (those containing the element) and negative patches (all other patches, including those from other classes).

To ensure scalability and fast inference—especially when processing large batches of images—GPU-accelerated libraries such as cuML(Team, 2023) are employed for algorithms like K-Means and Random Forest, significantly reducing classification time compared to CPU implementations. This is particularly important for applications requiring real-time or large-scale analysis, as it allows for efficient deployment of patch-based classification pipelines.

By comparing a diverse set of lightweight classifiers under identical binary classification conditions, this study aims to identify which approach is most effective for few-shot detection of architectural elements in streetview imagery, while maintaining the computational efficiency necessary for practical use.

Chapter 4 – Experimental Setup

The experimental setup chapter provides an overview of the datasets, model configurations, classifier choices, and evaluation metrics used to assess the proposed few-shot building element detection methodology. This setup is designed to enable reproducibility and provide justification for each choice within the context of few-shot learning for architectural element detection.

4.1 Dataset Construction and Preparation

The methodology and evaluation strategy of this project required the construction of a custom streetview dataset, designed to support few-shot learning for building element detection. The dataset is composed of three distinct subsets, each serving a specific role in the experimental pipeline:

Training set : 19,085 unlabeled Google Street View images, randomly sampled from addresses across the Netherlands. This set provides the basis for self-supervised Vision Transformer pretraining and representation learning.

Example set : For each architectural element of interest (solar panel, dormer, chimney, roof window, roof ventilation, parapet, balcony), up to 21 bounding box–annotated and 10 polygon–annotated images are included. These examples represent the few-shot learning constraint and are used to extract positive and negative patch embeddings for classifier training.

Test set : At least 100 images per element of interest, each labeled with a binary indicator denoting the presence or absence of the element. The test set is balanced for each class, containing a minimum of 49 positive and 49 negative examples, totaling approximately 700 images across all elements.

The foundation of the dataset is a diverse collection of streetview images. This carefully constructed dataset supports both the development and rigorous evaluation of few-shot building element detection methods in a realistic, address-level context.

4.2 Backbone Model Configurations

Two primary Vision Transformer backbones are evaluated to assess the impact of specialized retraining versus general-purpose representations. The NeCo backbone has the same architecture as DINOv2 model, but is retrained using the Patch Neighbor Consistency Loss methodology, with configurations trained for both 10 and 50 epochs on the 20,000 Dutch streetview images. The architecture incorporates 4 register tokens and maintains a 768-dimensional embedding

space, following the architectural specifications established by (Darcet et al., 2024). The DINOv2 backbone serves as the foundation model, utilizing the pretrained weights on the large-scale LVD-142M dataset (Oquab et al., 2023), without additional domain-specific retraining. This configuration enables direct assessment of whether general-purpose self-supervised representations suffice for domain-specific architectural element detection, or whether specialized retraining provides measurable improvements.

4.3 Patch Classifier Configurations

A set of lightweight classifiers was selected to ensure efficient binary classification of patch embeddings, given the large number of patches per image and the need for rapid inference. The classifiers were configured as follows:

- K-Means: In the configurations of 300 and 500 clusters.
- MLP: Single hidden layer with 64 units, trained for 50 epochs.
- SVM: RBF kernel and standard settings in cuML library.
- Random Forest: 250 estimators, maximum depth of 5.
- Mixture of Experts (MLP and RF): 10 experts initialized with experts being set up like the MLP and SVM above.
- XGBoost: Standard implementation xgboost python library (T. Chen and Guestrin, 2016).

Hyperparameters for each classifier were selected based on a series of trial-and-error runs to balance performance and computational efficiency. To accelerate inference and ensure scalability for large-scale patch classification, GPU-enabled libraries such as cuML are used for K-Means and Random Forest implementations. This configuration allows for fast and practical deployment of patch-based classification pipelines in real-world scenarios.

4.4 Evaluation Metrics: Average Accuracy and Precision (AAP)

In the domain of this project evaluation of a final model has to be done on two critical criteria: precision and overall performance. Precision ensures that when a positive prediction is made, it is likely correct, thereby minimizing false positives (Davis and Goadrich, 2006). However, relying solely on precision can lead to overly conservative models that miss many true instances. To address this, the F1 score is often used. It is defined as the harmonic mean of precision and recall, and is a metric of positive class performance:

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{4.1}$$

The F1 score excludes true negatives (TN), offering no insight into the model's ability to correctly identify negative cases. An F1 score can remain unchanged even when negative class performance deteriorates (Opitz, 2024). Balanced accuracy offers a more holistic view of model performance by incorporating both sensitivity and specificity:

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2} = \frac{\text{TPR} + \text{TNR}}{2} \tag{4.2}$$

Balanced accuracy gives equal weight to both positive and negative class performance, making it especially valuable for imbalanced datasets (Brodersen et al., 2010).

To evaluate overall model quality and precision in a single metric, this project introduces the Average Accuracy and Precision (AAP) metric. Defined as the arithmetic mean of balanced accuracy and precision, AAP combines reliable positive predictions and accurate classification across positive and negative classes:

$$\text{AAP} = \frac{\text{Balanced Accuracy} + \text{Precision}}{2} \tag{4.3}$$

This metric provides a simple basis to combine two important aspects of model performance: balanced accuracy and precision. AAP is used to streamline model selection with one number. However, for a more complete understanding of model behavior, the individual metrics, precision, recall, and balanced accuracy, are also reported and discussed in relevant sections of chapter 5.

Chapter 5 – Results

This chapter presents a structured evaluation of results of different Vision Transformer backbone and patch classifier combinations for few-shot building element detection. The findings are organized into the sections of: a comparison of backbone-classifier configurations, performance analysis when varying the numbers of few-shot examples, per element of interest evaluation of the best performing model, an ablation study examining critical pipeline components, and qualitative assessment of model predictions. The evaluation demonstrates that strategic combinations of feature extraction methods and classification approaches significantly impact performance in few-shot learning scenarios for building element detection tasks.

5.1 Vision Transformer & Patch Classifier Evaluation

The results for different Backbone + Patch Classifier configurations are presented in Table 5.1. This table shows performance metrics for setups using 21 examples per element of interest, with average scores reported across all elements. The highest performing configuration is the DINOv2 backbone paired with the RF-MoE patch classifier, achieving an AAP of 0.969. This result demonstrates the effectiveness of combining DINOv2's robust feature representations with the ensemble capabilities of Random Forest Mixture of Experts for patch embedding classification.

Baseline vs Others The results reveal several key insights about backbone and classifier combinations. Among the baseline NeCo configurations, the MLP patch classifier consistently outperforms clustering-based approaches, achieving the highest balanced accuracy (0.931) and AAP (0.924) in the baseline category. However, multiple backbone + classifier configurations outside of the baseline surpass its performance, with seven different setups exceeding the best baseline AAP score.

Simple Patch Classifiers vs Others Simple clustering-based classifiers (K-means with 300 and 500 clusters) generally underperform compared to more sophisticated classification methods across both backbones. The K-means approaches achieve modest AAP scores ranging from 0.807 to 0.880, while advanced classifiers like RF-MoE, SVM, and MLP consistently deliver superior performance. This pattern suggests that the rich feature representations from both NeCo and DINOv2 backbones require more complex classification strategies to fully exploit their discriminative capabilities. Notably, ensemble methods and traditional machine learning classifiers demonstrate significantly better precision-recall trade-offs compared to simple clustering approaches.

NeCo vs DINOv2 Backbone The DINOv2 backbone shows remarkable versatility across different classifiers. The RF-MoE configuration not only achieves the highest overall AAP but also maintains strong recall (0.922), indicating robust performance across all evaluation metrics. The standard Random Forest classifier with DINOv2 achieves perfect precision (1.000) but

Table 5.1: **Baseline**: NeCo backbone with k-means (300 and 500 cluster) or MLP patch classifiers. The baseline follows a similar implementation as (Pariza et al., 2025) and is compared to **Ours**: DINOv2 backbone with various patch classifiers. The baseline is set up like (Pariza et al., 2025) has done. The average performance scores are reported from a setup with 21 examples per 7 elements of interest.

	Backbone	**Patch Classifier**	**Precision**	**Recall**	**Bal Acc**	**AAP**
Baseline	NeCo	K-means 300	0.849	0.962	0.902	0.876
	NeCo	K-means 500	0.850	**0.976**	0.909	0.880
	NeCo	MLP	**0.917**	0.950	**0.931**	**0.924**
Ours	NeCo	RF	0.995	0.830	0.926	0.961
	NeCo	SVM	0.950	0.950	0.950	0.950
	NeCo	RF-MoE	0.956	0.910	0.936	0.946
	NeCo	XGBoost	0.821	0.948	0.882	0.852
	NeCo	MLP-MoE	0.847	0.973	0.905	0.876
	DINOv2	K-means 300	0.800	**0.989**	0.891	0.846
	DINOv2	K-means 500	0.750	0.984	0.863	0.807
	DINOv2	MLP	0.920	0.963	0.940	0.930
	DINOv2	RF	**1.000**	0.812	0.928	0.964
	DINOv2	SVM	0.944	0.974	**0.958**	0.951
	DINOv2	RF-MoE	0.982	0.922	0.956	**0.969**
	DINOv2	XGBoost	0.806	0.966	0.880	0.843
	DINOv2	MLP-MoE	0.802	0.981	0.888	0.845

with lower recall (0.812), suggesting a more conservative classification approach. Conversely, the SVM classifier provides the best balanced accuracy (0.958) while maintaining competitive performance across other metrics.

Interestingly, clustering-based approaches (K-means with 300 and 500 clusters) show contrasting behavior between backbones. While NeCo with K-means achieves reasonable performance (AAP of 0.876 and 0.880), DINOv2 with K-means demonstrates notably lower AAP scores (0.846 and 0.807) despite maintaining high recall rates. This suggests that DINOv2's patch feature space requires more sophisticated classification approaches to fully take advantage of its representational capabilities. It provides a stronger foundation for more complex classifiers. This is evidenced by the superior performance of ensemble methods like RF-MoE and traditional machine learning classifiers like SVM when paired with DINOv2 rather than NeCo.

5.2 Evaluation per Number of Examples

To assess the scalability and data efficiency of different backbone-classifier combinations, we evaluate performance across varying numbers of training examples per element of interest, as illustrated in Figure 5.1. This figure displays different ViT + Patch Classifier combinations, where different ViTs are indicated through line style and different patch classifiers through line color. The figure plots number of examples on the horizontal axis against AAP score on the vertical axis. Note that two versions of the NeCo ViT are included: one trained for 10 epochs and another trained for 50 epochs on the 20 000 Dutch housing images.

Observable Patterns As illustrated in Figure 5.1, the overall trend demonstrates that performance scores increase with additional example images. However, there is a notable anomaly

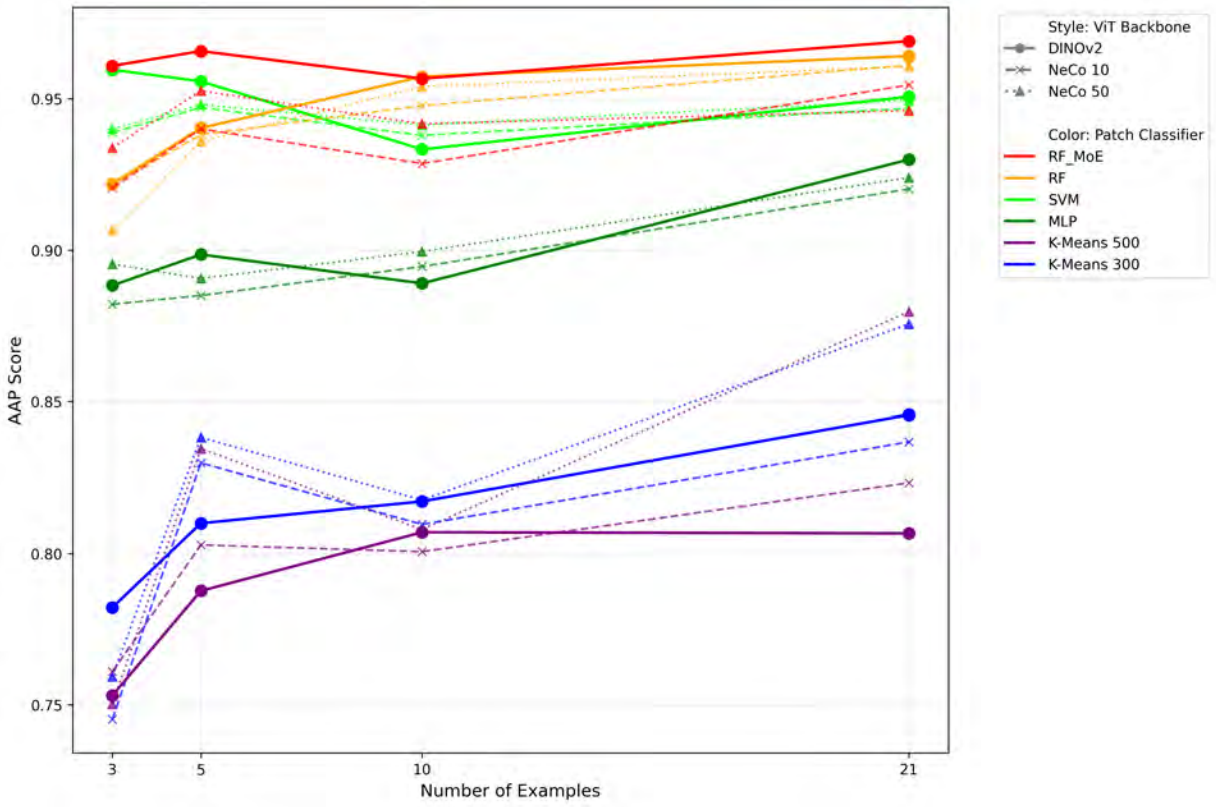


Figure 5.1: Per-model resulting Average Accuracy and Precision (AAP) versus Number of Examples are visualized. For a range of ViT backbones (indicated through line style) and Patch classifier (indicated through line color) combinations, results are plotted with AAP on the horizontal axis and the number of examples on the vertical axis. The scaling of the vertical axis is adjusted to better highlight small differences.

where almost all configurations show higher performance at 5 examples compared to 10 examples, creating a characteristic dip in the performance curves. This counterintuitive pattern likely results from the inclusion of particularly high-quality examples in the 5-example subset, while the expansion to 10 examples introduces additional lower-quality samples that temporarily degrade performance before the benefits of increased data volume become apparent at higher example counts.

Another trend we observe is that the performance spread between different configurations decreases as the number of examples increases from 3 to 21. The DINOv2 with RF-MoE model demonstrates the strongest performance on all numbers of examples, even with only three examples, while K-means-based models show considerably weaker results in low-data regimes. The K-means approaches only achieve acceptable performance levels when paired with NeCo-50 and provided with the full 21 examples. This pattern highlights the critical importance of selecting robust backbone-classifier combinations for few-shot learning scenarios, where data scarcity impacts performance differently across classifier types.

Together with the results in Table 5.1, it must be concluded that the overall best performing model is the DINOv2 + RF-MoE model, as it achieves the best balance of precision and balanced accuracy in terms of AAP.

5.3 Evaluation per Element of Interest

After selecting the best model based on overall performance, the DINOv2 + RF-MoE configuration, we examine its performance per element of interest as presented in Table 5.2. This analysis serves as an important step in gaining a deeper understanding of the methodology's behavior and capabilities. The results are based on approximately 50 positive and 50 negative test images per class, totaling around 700 images.

The per-class analysis reveals significant performance variation across different building elements, closely correlating with their visual characteristics and detectability in street-view imagery. The model achieves perfect performance (1.000 AAP) for both dormers and roof windows, indicating that DINOv2 patch embeddings from these elements are highly distinct and the RF-MoE can easily differentiate between patch embeddings belonging to these classes versus those not belonging to these classes. Chimneys also demonstrate excellent performance with 0.995 AAP, showing near-perfect precision (1.000) with only minor recall limitations. These elements share the common characteristic of being relatively large, so that they remain clearly visible and recognizable even in 640×640 street-view images.

The most challenging element proves to be roof ventilation, achieving the lowest performance across all metrics (0.896 AAP, 0.732 recall). This significant performance drop can be attributed to the visual characteristics of roof ventilation systems, which appear as small black tubes or vents on rooftops. In 640×640 street-view images, these elements are often small and highly pixelated, making detection challenging. A roof ventilation element, being small in size, often does not cover more than one patch, and when the element falls on the boundaries of multiple patches, this results in an even lower chance of accurately identifying those patches as belonging to that class. The low recall (0.732) compared to relatively high precision (0.953) indicates that while the model frequently misses instances caused by the small size of these features, it correctly identifies roof ventilation when detected.

Overall, the per-element analysis demonstrates that the methodology performs exceptionally well for architecturally distinct elements with sufficient size and visual contrast (dormers, roof windows, chimneys, solar panels), while facing greater challenges with smaller or more subtle features (roof ventilation) and elements that may be confused with similar architectural components (parapets). This performance distribution aligns with the limitations of patch-based classification in street-view imagery, where element size and visual distinctiveness are critical factors for successful detection.

Table 5.2: Best Ours based on AAP score: DINOv2 RF-MoE per class performance.

Category	**Precision**	**Recall**	**Bal Acc**	**AAP**
Solar Panel	0.981	0.981	0.980	0.981
Dormer	**1.000**	**1.000**	**1.000**	**1.000**
Chimney	**1.000**	0.981	0.990	0.995
Roof Window	**1.000**	**1.000**	**1.000**	**1.000**
Roof Ventilation	*0.953*	*0.732*	*0.856*	*0.896*
Parapet	0.976	0.800	0.906	0.941
Balcony	0.963	0.982	0.961	0.962
Mean	**0.982**	**0.922**	**0.952**	**0.967**

5.4 Ablation study

To understand the individual and combined contributions of the methodology's key components, a systematic ablation study focusing on three critical design choices is conducted: annotation style, resampling strategy, and adaptive cropping. These components were selected based on their potential impact on patch quality and class balance in few-shot learning scenarios. The experiments were conducted with 10 examples per element of interest instead of 21, due to time constraints and the time-consuming task of producing polygon annotations. The results of this ablation study are presented in Table 5.3 and will be discussed in the next paragraphs.

Table 5.3: Ablation study setup and results for the example-patches collection strategy. The results were obtained using a DINOv2 backbone with the RF-MoE patch classifier with 10 examples per class. The performance scores are given as an average over the 7 elements of interest.

Ablation setup			**Performance Scores**		
Annotation Style	**Resampling**	**Adaptive Crop**	**Precision**	**Balanced Accuracy**	**AAP**
Polygon $t = 0.7$	✓	-	0.962	0.919	0.941
Polygon $t = 0.7$	×	-	0.838	0.869	0.853
Bounding Box	✓	✓	0.971	**0.943**	**0.957**
Bounding Box	✓	×	0.569	0.737	0.653
Bounding Box	×	✓	**0.975**	0.926	0.950
Bounding Box	×	×	0.569	0.737	0.653

Annotation Style The annotation style significantly impacts performance across all metrics. Polygon annotations achieve competitive results when combined with resampling (0.919 balanced accuracy, 0.941 AAP) but show substantial degradation without this preprocessing step (0.853 AAP). Note that polygon annotations are not supported with an adaptive cropping implementation due to their irregular shape. Bounding box annotations demonstrate the highest potential, yielding the best overall performance when combined with adaptive cropping (0.957 AAP). However, bounding box annotations are highly dependent on the adaptive cropping methodology, such that without it, they underperform significantly (0.653 AAP).

Resampling Resampling involves undersampling negative patches to maintain at least 1 % positive samples of the element of interest in the training set. The results demonstrate that resampling provides substantial benefits for polygon annotations, improving AAP from 0.853 to 0.941, indicating its effectiveness in addressing class imbalance issues. However, for bounding box annotations, resampling shows mixed results depending on the presence of adaptive cropping. When adaptive cropping is absent, resampling maintains consistent performance (0.653 AAP in both cases), while with adaptive cropping enabled, resampling provides only marginal improvements (0.957 vs 0.950 AAP). Looking back at Table 3.1, the bounding box annotation style without adaptive cropping yields minimal patch counts for certain elements of interest. In these cases, the absolute lack of positive patches cannot be adequately addressed through resampling alone, as the fundamental issue remains the insufficient number of training examples rather than class imbalance.

Adaptive Cropping Adaptive cropping demonstrates the strongest individual impact on performance. For bounding box annotations, it provides dramatic improvements, increasing

AAP from 0.653 to 0.950 when used without resampling, and from 0.653 to 0.957 when combined with resampling. This substantial improvement occurs because adaptive cropping addresses the fundamental issue of insufficient positive patches by actively including regions containing the elements of interest, thereby increasing the absolute number of training examples rather than merely adjusting class ratios.

Key Findings The ablation study reveals that preprocessing strategies are crucial for few-shot architectural element detection. Adaptive cropping for bounding box annotations emerges as the most impactful component, while resampling provides complementary benefits for polygon annotations. The optimal configuration combines bounding box annotations with both adaptive cropping and resampling, achieving 0.957 AAP. These findings highlight the importance of addressing both patch quality (through adaptive cropping) and class balance (through resampling) in few-shot learning scenarios.

5.5 Qualitative Results

In this section, we present qualitative results by inspecting test set images on which inference was performed. The images are displayed with a mask overlay that highlights, in red, the patches where the model has detected the element of interest. To avoid overloading this section with images, only the first three elements of interest, "solar panel" (Figure 5.2), "dormer" (Figure 5.3), and "chimney" (Figure 5.4), will be discussed here. Qualitative results for the remaining four elements are included in section A.1, along with a brief analysis for each. For every element of interest, a figure is provided with up to 6 subfigures (if available). For every element of interest, a figure is provided with up to 6 subfigures (if available). Per element of interest, the top row of images provide qualitative results for the first model: the *NeCo + K-means 300* (baseline) model, and in the bottom row, the best-performing model: *DINOv2 + RF-MoE* (our) from Table 5.1. This setup enables direct comparison of model performance. In each row, a "True Positive", "False Positive", and "False Negative" example is shown. For each case, the first test image is used to prevent cherry-picking. "True Negatives" are not included, as they offer limited value for qualitative evaluation. A true negative represents an image without the element of interest, correctly ignored by the model. Although all configurations identify many true negatives, these contribute little to understanding model behavior or failure modes.

Qualitative Results for Solar Panel Detection Solar panels are relatively large architectural elements that are therefore clearly visible when present in images captured from reasonable distances. Their presence in any given image is influenced by several factors including the viewing angle, roof configuration and orientation, as well as their positioning, which determine whether they are visible from the street perspective.

The true positive patches in the baseline model are less frequent than in our model, where certain image areas containing solar panels are not identified as such. Our model occasionally includes patches that contain only a small portion of a solar panel, indicating that the model employs a less restrictive boundary for solar panel patch embeddings, though both the baseline and our approach only highlight patches where solar panels are actually present.

The false positive in the baseline model incorrectly flags a region as containing a solar panel, which has no visual similarity to solar panels beyond the fact that solar panels typically appear in the upper portions of buildings. Our model produces only one false positive, a black

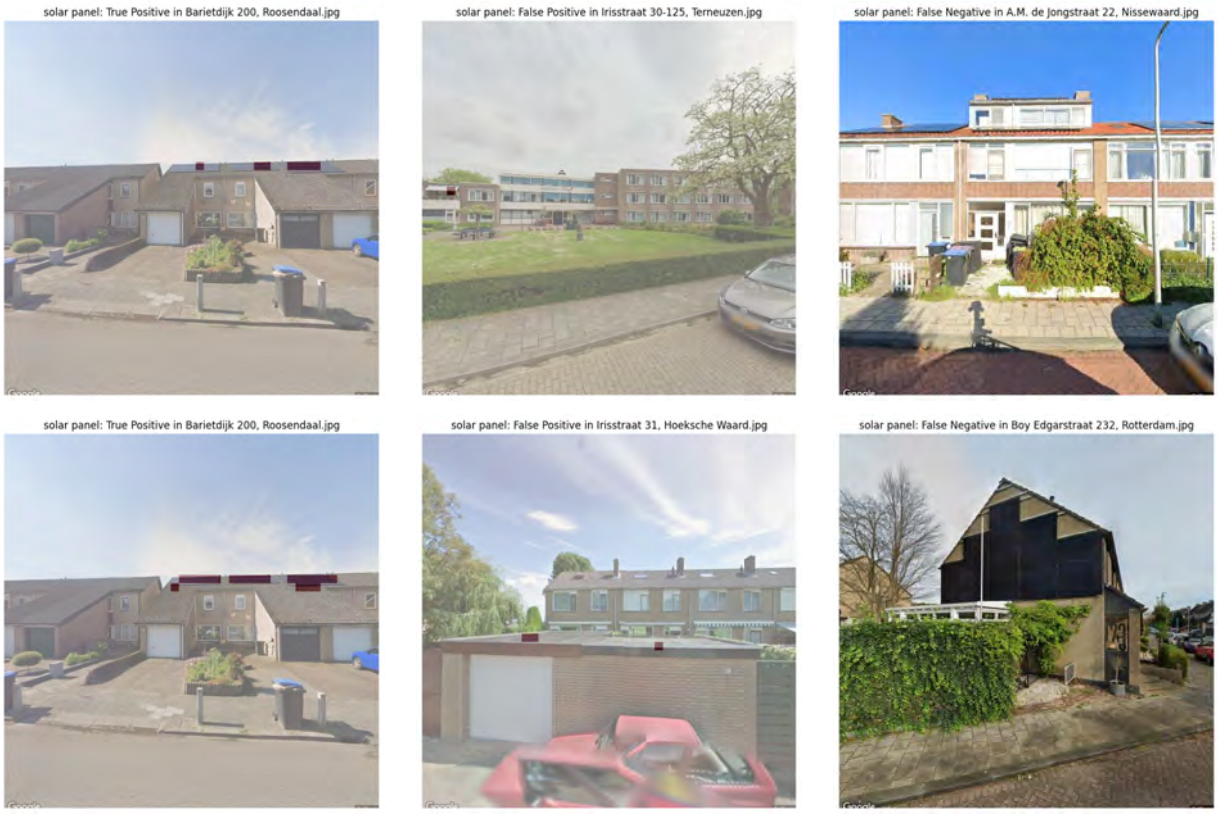


Figure 5.2: Qualitative results for **Solar Panel**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

tar roof reflecting light in a manner similar to that commonly observed on solar panels. This same error occurs in the baseline model, but is not visualized here as the first image in the sequence of test images flagged as false positive by the baseline appears before our false positive.

The false negative in the baseline clearly displays solar panels that should have been detected. It shows a standard image of a house with solar panels clearly present on the roof. Our model again has only one false negative, which is an image where the solar panel is positioned in a very unusual manner: mounted on the side of a house wall. Due to this unconventional placement, even a human observer must closely inspect the image to determine with certainty whether this is actually a solar panel or not.

Qualitative Results for Dormer Detection Dormers are also relatively large architectural elements. Their greater volume compared to solar panels makes them more prominent, and although they may not be present in a streetview image when located at the back of the house or under specific perspectives, the probability of capturing them in a streetview image when present is expected to be higher.

Inspecting the true positives highlighted by the baseline and our model reveals that again, our model appears to have a more relaxed fit for dormer patches. It wrongfully highlights a patch that does not actually contain the dormer, a patch of blue sky. The baseline model demonstrates a more precise fit of the dormer in this example, concisely highlighting patches that encompass the dormer.

The example of a false positive by the baseline shows a patch on a house that is in the correct

* * * *No FP available here* * * *

* * * *No FN available here* * * *

Figure 5.3: Qualitative results for **Dormer**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

spatial region of the house, but clearly does not display a dormer. This patch should not have been flagged as positive and our model correctly does not do this.

In the background of the false negative from the baseline's dormer prediction, a dormer can be observed. This is a more challenging image, and may not be important for the use case of this project, given the focus on the foreground of streetview images. However, when compared to our model, which does not have any false negatives, this means that it has correctly identified the dormer in this image.

Qualitative Results for Chimney Detection Chimneys are small architectural elements in streetview images. However, due to their location at the top of a roof, they are almost always visible in a streetview image when present on a house.

The true positives of the baseline and our model do not have very noteworthy differences, as the only distinction is one additional patch highlighted as a chimney in the baseline model with only a small portion of the patch actually containing a chimney.

The false positive displayed by the baseline model is a difficult case, where the object flagged resembles a chimney, but it is not a classic brick one. Close inspection reveals that it may be a ventilation shaft. If it were a chimney, this type of chimney is not present in the example set, which only consists of brick chimneys, and the aim is, to have a model that searches for the specific examples provided to the model.

The false negatives by the baseline and our model are both examples of images with small

Figure 5.4: Qualitative results for **Chimney**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

chimneys in the background of the image. These cases show the inherent difficulty in detecting small architectural elements that appear at a distance, where limited pixel resolution and prominence make accurate detection difficult in both models.

Overall, it can be concluded that the baseline model exhibits more significant failures than our model across all architectural elements discussed in this section. Our model appears to have learned a more robust and precise representation of what each element of interest visually looks like, showing better adherence to the specific examples provided during training. Other than that, it becomes clear that with qualitative inspection, the approach of this thesis is highly interpretable. Overlaying an image with masks of classified patches from ViT + Classifier combinations clearly visualizes whether a good representation of an element of interest was learned or not.

Chapter 6 – Conclusion

This thesis successfully demonstrates that self-supervised Vision Transformers combined with few-shot learning provide an effective solution for building element detection in streetview imagery. The DINOv2 backbone paired with Random Forest Mixture of Experts (RF-MoE) achieves 0.969 AAP, detecting seven different building elements using only 3-21 labeled examples per element.

The research establishes three primary contributions. First, DINOv2's patch embeddings prove superior to the specialized retraining approaches like NeCo for domain-specific detection tasks when paired with a complex classifier. Second, adequate labeling strategies are key in ensuring high performance scores. Together with the right patch selection mechanism, it ensures proper alignment between annotations and the ViT's patch grid, increasing AAP from 0.653 to 0.950. Third, the RF-MoE classifier effectively exploits the rich representations from self-supervised ViTs, outperforming simple clustering approaches across all evaluation metrics.

The methodology directly supports TNO's Clustertool and the Netherlands' contingentenaanpak initiative, providing a scalable solution for enriching address-level building data. Results on the impact of integrating the obtained features in the TNO Clustertool are presented and discussed in A.1. The ViT + Classifier approach achieves reliable detection with minimal supervision, making it practically deployable without expansive annotation costs. Performance analysis reveals excellent results for certain building elements (perfect scores for dormers and roof windows) while highlighting challenges with smaller features like roof ventilation.

The patch-based approach faces inherent limitations with very small architectural elements that may not span complete 14×14 pixel patches. Future research could explore multi-scale patch processing.

This work validates the practical value of self-supervised Vision Transformers in specialized domains with limited labeled data, contributing to both computer vision research and climate action initiatives. By enabling automated identification of renovation-relevant building features, the methodology supports the Netherlands' ambitious goal of renovating 1.5 million homes by 2030, demonstrating how a computer vision technique can ultimately lead to positive environmental impact.

Bibliography

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning*. Ed. by M. Jordan, J. Kleinberg, and B. Schölkopf. Information Science and Statistics. New York, NY: Springer Science+Business Media, LLC. ISBN: 0-387-31073-8.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45, pp. 5–32.

Brodersen, Kay Henning et al. (2010). "The balanced accuracy and its posterior distribution". In: *2010 20th international conference on pattern recognition*. IEEE, pp. 3121–3124.

Caron, Mathilde et al. (2021). *Emerging Properties in Self-Supervised Vision Transformers*. arXiv: 2104.14294 [cs.CV]. URL: https://arxiv.org/abs/2104.14294.

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 785–794. DOI: 10.1145/2939672.2939785.

Darcet, Timothée et al. (2024). *Vision Transformers Need Registers*. arXiv: 2309.16588 [cs.CV]. URL: https://arxiv.org/abs/2309.16588.

Davis, Jesse and Mark Goadrich (2006). "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240.

Dosovitskiy, Alexey et al. (2021). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

Hamilton, Mark et al. (2022). *Unsupervised Semantic Segmentation by Distilling Feature Correspondences*. arXiv: 2203.08414 [cs.CV]. URL: https://arxiv.org/abs/2203.08414.

Hao, Fusheng et al. (2023). "Class-aware patch embedding adaptation for few-shot image classification". In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 18905–18915.

He, Kaiming, Xinlei Chen, et al. (2021). *Masked Autoencoders Are Scalable Vision Learners*. arXiv: 2111.06377 [cs.CV]. URL: https://arxiv.org/abs/2111.06377.

He, Kaiming, Xiangyu Zhang, et al. (2015). *Deep Residual Learning for Image Recognition*. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

Jiang, Weihao, Haoyang Cui, and Kun He (2024). "Class-relevant Patch Embedding Selection for Few-Shot Image Classification". In: *arXiv preprint arXiv:2405.03722*.

Kuhn, Harold W (1955). "The Hungarian method for the assignment problem". In: *Naval research logistics quarterly* 2.1-2, pp. 83–97.

Liu, Shilong et al. (2024). *Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection*. arXiv: 2303.05499 [cs.CV]. URL: https://arxiv.org/abs/2303.05499.

Nielsen, Stefan K. et al. (2025). *Tight Clusters Make Specialized Experts*. arXiv: 2502.15315 [cs.LG]. URL: https://arxiv.org/abs/2502.15315.

Niu, Dantong et al. (June 2024). "Unsupervised Universal Image Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22744–22754.

Opitz, Juri (2024). "A closer look at classification evaluation metrics and a critical reflection of common evaluation practice". In: *Transactions of the Association for Computational Linguistics* 12, pp. 820–836.

Oquab, Maxime et al. (2023). "Dinov2: Learning robust visual features without supervision". In: *arXiv preprint arXiv:2304.07193*.

Pariza, Valentinos et al. (Mar. 2025). "Near, far: Patch-ordering enhances vision foundation models' scene understanding". In: *The Thirteenth International Conference on Learning Representations*. URL: https://vpariza.github.io/NeCo/.

Pearson, Karl (1901). "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin philosophical magazine and journal of science* 2.11, pp. 559–572.

Rambhatla, Sai Saketh et al. (2023). *MOST: Multiple Object localization with Self-supervised Transformers for object discovery*. arXiv: 2304.05387 [cs.CV]. URL: https://arxiv.org/abs/2304.05387.

Siméoni, Oriane et al. (2021). *Localizing Objects with Self-Supervised Transformers and no Labels*. arXiv: 2109.14279 [cs.CV]. URL: https://arxiv.org/abs/2109.14279.

Strano Moraes, Luis Felipe et al. (2025). "Applying Vision Transformers on Spectral Analysis of Astronomical Objects". In: *arXiv e-prints*, arXiv–2506.

Team, RAPIDS Development (2023). *RAPIDS: Libraries for End to End GPU Data Science*. URL: https://rapids.ai.

Appendix A – Appendix

A.1 More Qualitative Results

Qualitative Results for Roof Window Detection The qualitative analysis of roof window detection highlights clear differences in performance between the K-means baseline and the RF-MoE classifier. The K-means model often fails to identify all patches corresponding to the roof window, missing several regions that contain parts of the element. Additionally, it incorrectly flags unrelated areas on the roof as containing a roof window, resulting in false positives. Notably, the K-means classifier also overlooks background roof windows that are successfully detected by the RF-MoE model.

In contrast, the RF-MoE classifier demonstrates more precise localization, accurately covering the majority of patches that contain the roof window. It is able to identify both prominent and background roof windows, indicating a more reliable detection. Overall, these results illustrate that RF-MoE provides superior patch-level discrimination for roof window detection, reducing both missed detections and false activations compared to the K-means approach.

Qualitative Results for Roof Ventilation Detection The qualitative results for roof ventilation detection reveal distinct behaviors between the K-means and RF-MoE classifiers. The K-means classifier tends to over-predict, labeling many patches as positive for roof ventilation even when they do not correspond to actual roof vents. This results in numerous false positives. The model does not produce any false negatives, suggesting that its definition of roof ventilation is overly broad and lacks specificity.

In contrast, the RF-MoE classifier is more selective, typically identifying only the true roof ventilation patches in true positive cases. However, in the false positive example, it incorrectly classifies the top part of a chimney as a roof vent—likely due to visual similarity between these elements. The RF-MoE classifier shows a conservative approach, occasionally failing to detect roof vents even when they are present, resulting in false negatives. This limitation is likely due to the small size of roof ventilation elements in street view images, which makes them more challenging properly cover in the ViTs patches.

Figure A.1: Qualitative results for **Roof Window**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

Figure A.2: Qualitative results for **Roof Ventilation**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

Qualitative Results for Parapet Detection In parapet detection, both the K-means and RF-MoE classifiers perform similarly in true positive cases, successfully identifying parapets when they are clearly visible. In the false positive examples, the K-means classifier is confused by a visually similar structure: a window shade is flagged as a parapet, likely due to its similar appearance from a distance. The RF-MoE incorrectly identifies a brick wall as a parapet. The K-means model also fails to detect parapets in certain apartment-style buildings, which may be due to underrepresentation of such examples in the annotation set. In contrast, the RF-MoE sometimes misses parapets that are partially obscured by objects like plants or cars, indicating sensitivity to occlusion.

It is important to note that in several images where the K-means classifier made errors, the RF-MoE correctly identified the parapet. However, these cases are not shown in the qualitative results, as the first misclassified images in the alphabetic test set were selected for visualization. This highlights that while both models have specific failure modes, the RF-MoE often succeeds where K-means fails, particularly in more challenging or ambiguous cases.

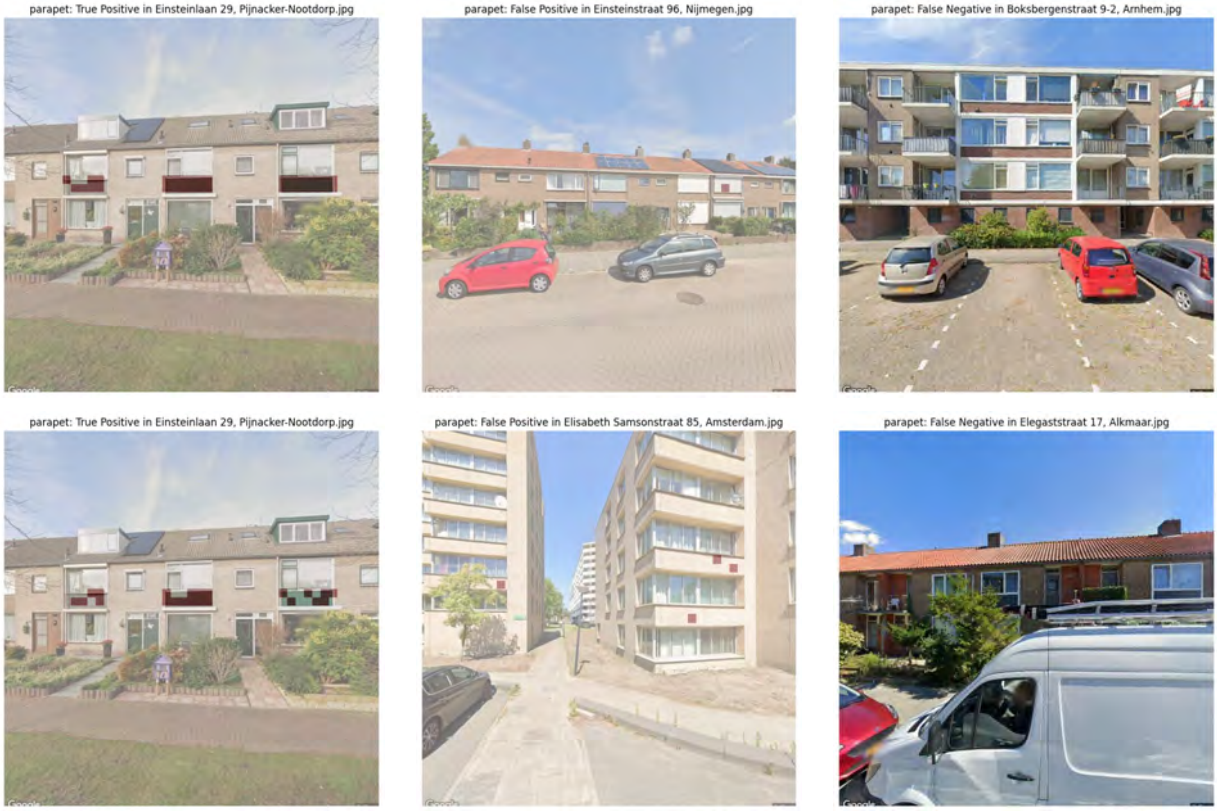


Figure A.3: Qualitative results for **Parapet**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

Qualitative Results for Balcony Detection The qualitative assessment of balcony detection reveals differences and challenges for both classifiers. In the true positive example, the K-means classifier tends to classify more patches as balconies compared to the RF-MoE, indicating a less selective approach.

In the false positive scenario, K-means incorrectly classifies the top of a roof segment as a balcony, while the RF-MoE mistakenly identifies a piece of railing attached to a building as a balcony. For the false negative case, the K-means classifier fails to detect a balcony in the

background, and the RF-MoE does not identify a typical balcony present in the image.

These observations underscore the diversity of balcony types and appearances across different housing styles, which poses a significant challenge for both models. The variability in balcony design suggests that further separating different balcony styles in annotation could improve detection performance when targeting specific types.

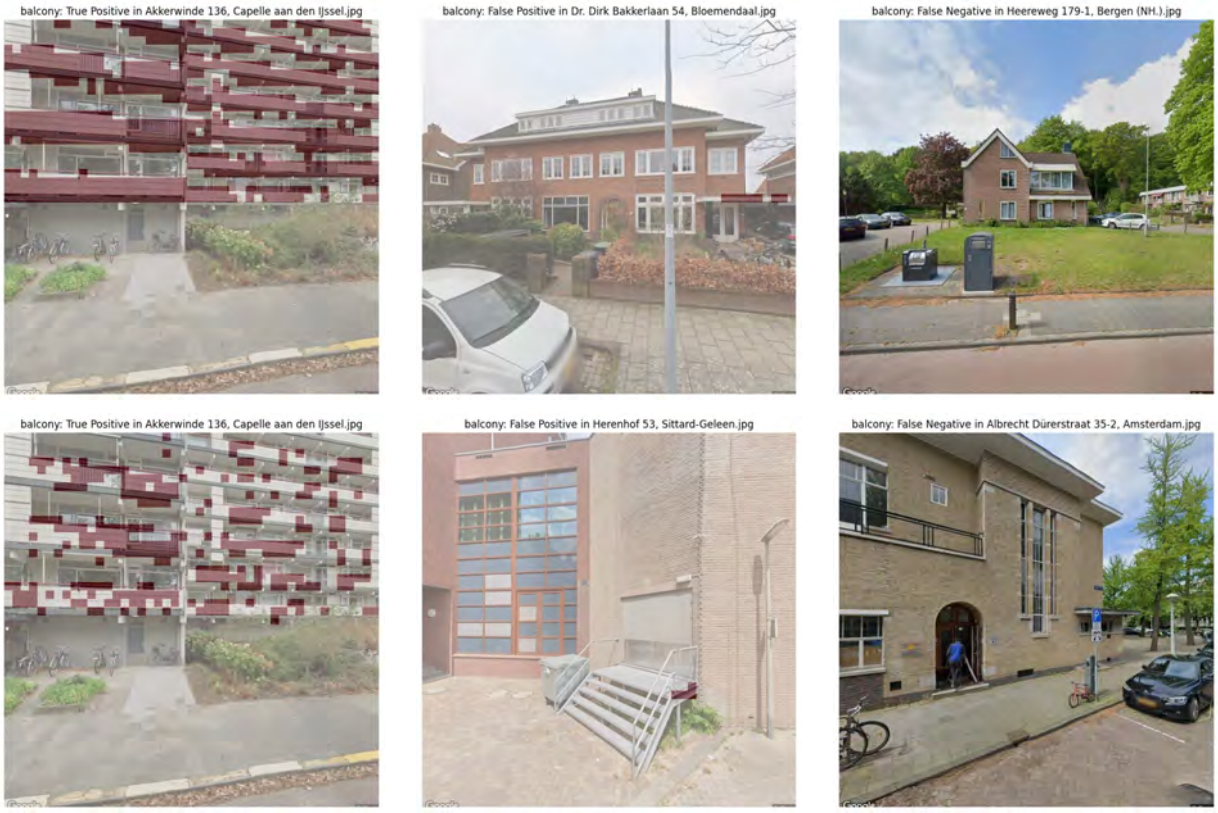


Figure A.4: Qualitative results for **Balcony**. The **columns** of this figure display: **TP / FP / FN**, the **top row** displays the results for **NeCo + K-means300** and the **bottom row** shows results for **DINOv2 + RF-MoE** patch classification.

A.2 TNO Clustertool Results

The TNO Clustertool evaluation in Table A.1 reveals important insights about how visual features from streetview imagery impact renovation solution prediction accuracy. The results demonstrate varying effectiveness of implicit features (PCA-reduced DINOv2 CLS tokens), explicit features (detected building elements with DINOv2 + RF-MoE), and their combination across different renovation solutions.

The most substantial improvements occur for renovation solutions with high visual correlation to detectable building elements. Roof Insulation shows a large explicit feature improvement (+4.8 percentage points), reaching 63.4 % balanced accuracy. This aligns with the roof-related element detection (dormers, roof windows). Similarly, Pitched Roof Insulation benefits significantly from explicit features (+1.7 percentage points), demonstrating that the many roof-related elements provide valuable additional information towards the improvement of the Clustertool.
Parapet detection yields particularly interesting results. For Parapet House classification, explicit features achieve the best performance (78.8 %, +5.9 percentage points), confirming that

Table A.1: Balanced accuracy of the TNO Clustertool in predicting the applicability of renovation solutions. *Img Features* indicates which image inputs are included: *none* (no image features), *implicit* (PCA-reduced DINOv2 CLS token from street view images), *explicit* (seven elements of interest from street view images), or *combined* (implicit and explicit). *Balanced accuracy* is shown with its margin of error, and the top score per renovation solution is in bold. The *Significant* column flags statistically significant changes due to visual features; the *Change* column reports magnitude of significant change in percentage points.

Renovation Solution	**Img Features**	**Bal Acc**	**Significant**	**Change**
Solar Panels	none	**64.2** ± 0.1		
	implicit	63.2 ± 0.1	YES	-1
	explicit	**64.2** ± 0.1	NO	
	combined	63.3 ± 0.1	YES	-0.9
Insulating Glass	none	53.7 ± 0.2		
	implicit	52.5 ± 0.2	YES	-1.2
	explicit	**53.8** ± 0.2	NO	
	combined	52.7 ± 0.2	YES	-1
Floor Insulation	none	**62.3** ± 0.1		
	implicit	61.4 ± 0.1	YES	-0.9
	explicit	62.2 ± 0.1	NO	
	combined	61.5 ± 0.1	YES	-0.8
Cavity Wall Insulation	none	63.1 ± 0.1		
	implicit	**63.5** ± 0.2	YES	**0.4**
	explicit	63.1 ± 0.1	NO	
	combined	63.2 ± 0.2	YES	0.1
Hybrid Heat Pump	none	56.6 ± 0.3		
	implicit	52.7 ± 0.2	YES	-3.9
	explicit	**56.8** ± 0.3	NO	
	combined	52.6 ± 0.2	YES	-4
Roof Insulation	none	58.6 ± 0.5		
	implicit	62.0 ± 0.5	YES	3.4
	explicit	**63.4** ± 0.6	YES	**4.8**
	combined	63.2 ± 0.5	YES	4.6
Pitched Roof Insulation	none	50.3 ± 0.4		
	implicit	50.3 ± 0.2	NO	
	explicit	**52.0** ± 0.4	YES	**1.7**
	combined	50. ± 0.3	NO	
Window Frames	none	**49.8** ± 1.1		
	implicit	49.7 ± 1.1	NO	
	explicit	49.7 ± 1.1	NO	
	combined	**49.8** ± 1.1	NO	
Parapet House	none	72.9 ± 0.6		
	implicit	77.6 ± 0.6	YES	4.7
	explicit	**78.8** ± 0.5	YES	**5.9**
	combined	78.7 ± 0.6	YES	5.8
Parapet Apartment	none	55.8 ± 1.9		
	implicit	**67.4** ± 1.7	YES	**11.6**
	explicit	56.6 ± 1.5	NO	
	combined	66.6 ± 1.6	YES	10.8

direct parapet detection from streetview imagery effectively identifies relevant building types. However, Parapet Apartment classification shows a different pattern—implicit features perform best (+11.6 percentage points), while explicit features show no significant improvement. This is also in line with expectations, as in the labeling process, no examples of parapets on apartment buildings were provided.

Several renovation solutions show minimal or negative impact from visual features. Solar Panel prediction accuracy remains unchanged with explicit features (64.2%), despite successful solar panel detection in the few-shot experiments. This may be attributed to the fact that a streetview image may detect a solar panel on the roof of a neighboring house. This issue does not arise when detecting parapets, which is an element that is usually also present in entire streets when present in one of the buildings.

The comparison between implicit and explicit features reveals task-dependent effectiveness. For renovation solutions directly related to detectable building elements (roof insulation, parapet houses), explicit features generally outperform implicit features. However, for more finegrained architectural classifications (Cavity Wall) or solutions requiring assessment of building characteristics not captured by the seven explicit elements, implicit features from DINOv2's general representations prove more valuable. Cavity Wall Insulation shows a modest but significant improvement with implicit features (+0.4 percentage points), suggesting that wall construction characteristics are better captured through general visual representations than through specific building element detection.

These results validate the practical value of integrating building element detection into renovation planning tools. The methodology demonstrates clear benefits for renovation solutions with strong visual correlations to detectable elements, while showing appropriate restraint (no false improvements) for solutions requiring non-visual assessment criteria. The neighbor-consistent performance for parapet detection supports the scalability of streetview-based approaches for neighborhood-level renovation planning.

The mixed results across renovation types provide valuable guidance for deployment priorities. Importantly, implicit features serve as an effective indicator of whether relevant visual information is present in streetview imagery, and when that implicit signal is explicitly captured through targeted element detection, performance typically increases further. This pattern suggests that implicit features can guide the development of explicit detection models by identifying which building characteristics are learnable from streetview images.