Deep Learning for Satellite Building Change Detection:

From Convolutional Baselines to Siamese Temporal Attention

Project Head:
Lorenzo **De Santis**

Team Members:
Sofia **Elisa**
Thomas **Agosti**

HEPHAESTUS APPLIED ARTIFICIAL INTELLIGENCE ASSOCIATION

Spring 2026

PROJECT REPORT

Abstract

Remote sensing change detection aims to identify meaningful differences between images acquired over the same geographic region at different times. In urban scenarios, the task is particularly relevant for monitoring construction, demolition, and land-use evolution, yet it remains challenging because true structural changes are often sparse and can be obscured by illumination variation, shadows, seasonal differences, and small registration errors. This paper presents a complete deep learning pipeline for binary building change detection on the LEVIR-CD dataset. The study compares three increasingly expressive architectures under a shared preprocessing, loss, optimization, and evaluation protocol: a shallow full-resolution convolutional baseline, an early-fusion U-Net, and a Siamese Temporal Attention U-Net. The baseline quantifies the performance achievable with local convolutional processing alone; the U-Net introduces multi-scale encoder-decoder segmentation; the Siamese Temporal Attention model explicitly encodes the two temporal images with shared weights and applies global attention at the bottleneck to model cross-time feature interactions. All models are trained on 256×256 patches with a weighted binary cross-entropy and Dice objective, and are evaluated using F1-score, intersection-over-union, Dice coefficient, precision, and recall. The final test results show a clear progression in performance: the simple CNN reaches a test F1-score of 0.7093, the U-Net reaches 0.8840, and the Siamese Temporal Attention U-Net reaches 0.9019. These results indicate that encoder-decoder structure is essential for dense satellite segmentation, while explicit temporal comparison further improves robustness and boundary-level discrimination. The paper also reports hyperparameter optimization, qualitative prediction maps, error visualizations, and implementation details needed for reproducibility.

Keywords: remote sensing, change detection, LEVIR-CD, convolutional neural networks, U-Net, Siamese networks, temporal attention, semantic segmentation.

Contents

1 Introduction

Remote sensing imagery provides a systematic and scalable way to observe changes in the Earth's surface. When the same location is imaged at different times, the resulting bitemporal pair can reveal construction, demolition, infrastructure expansion, vegetation changes, or damage following natural disasters. Among these applications, urban building change detection is particularly important because changes in the built environment are strongly connected to population growth, economic development, land consumption, and disaster response. The problem considered in this work is binary building change segmentation: given two co-registered satellite images acquired over the same geographic area at two different timestamps, the objective is to predict a dense binary mask indicating pixels where building-related changes occurred.

Although the formulation is simple, the task is technically demanding. A model must distinguish real semantic changes from nuisance variation. Bitemporal images may differ in illumination, seasonal appearance, shadow direction, local contrast, or acquisition conditions. In addition, small misregistration errors may generate apparent differences near unchanged building boundaries. The LEVIR-CD dataset paper emphasizes precisely these difficulties and motivates the need for methods that exploit relationships across spatial and temporal positions rather than treating the two images as independent observations (Chen & Shi, 2020). This project follows the same conceptual premise, but implements a compact and reproducible comparison among three neural architectures designed to clarify the contribution of progressively stronger inductive biases.

The first model is a simple convolutional neural network operating at full spatial resolution. It receives the two RGB images concatenated along the channel dimension and produces a one-channel logit map. This model functions as a controlled baseline: it tests whether a shallow local model can learn useful change cues without an encoder-decoder structure. The second model is an early-fusion U-Net. It also receives a six-channel input, but uses a multi-scale encoder-decoder with skip connections, which is a more suitable structure for dense segmentation. The third model is a Siamese Temporal Attention U-Net. Instead of concatenating the raw images at the input, it processes the two images through a shared encoder, applies global temporal attention at the bottleneck, fuses corresponding feature maps through temporal difference-aware blocks, and decodes the result into a change mask.

The contribution of this paper is threefold. First, it provides a clean experimental comparison among a convolutional baseline, an early-fusion segmentation network, and a Siamese temporal attention architecture on the same dataset and preprocessing pipeline. Second, it formalizes the temporal attention mechanism used in the final model and explains why it is aligned with the comparative nature of bitemporal change detection. Third, it reports a complete training and evaluation protocol, including hyperparameter optimization, early stopping, quantitative results, qualitative visualizations, and implementation details. The resulting comparison demonstrates that the final Siamese Temporal Attention U-Net improves test F1-score from 0.8840 for U-Net to 0.9019, while maintaining the same input resolution and training data.

The remainder of the paper is organized as follows. Section 2 introduces the relevant background and the position of this work with respect to LEVIR-CD. Section 3 describes the dataset, preprocessing, and exploratory data analysis. Section 4 presents the three architectures, loss function, metrics, and mathematical formulation of temporal attention. Section 5 reports the experimental results. Section 6 discusses the implications of the comparison. Section 7 concludes the paper, and Appendix A provides additional implementation details and qualitative examples.

2 Background and Related Work

2.1 Remote Sensing Change Detection

Remote sensing change detection concerns the identification of relevant differences between multi-temporal observations of the same region. In the binary formulation considered here, each pixel is assigned to either the unchanged class or the changed class. The target mask therefore encodes a semantic difference rather than a raw radiometric difference: a pixel should be marked as changed only when the underlying scene has changed according to the annotation policy. In LEVIR-CD, the annotations focus on building changes, and the data consist of bitemporal Google Earth images with manually labeled change instances (Chen & Shi, 2020).

The difficulty of the task arises from the fact that the observed image difference is not identical to the semantic change signal. Let $X^{(1)}$ and $X^{(2)}$ denote two images acquired at times t_1 and t_2. A naive difference image $|X^{(2)} - X^{(1)}|$ responds to any pixel-level discrepancy, including illumination, shadows, compression artifacts, vegetation appearance, and slight displacement. The desired output, however, is a binary function Y that marks only meaningful structural changes. A learning-based method must therefore suppress nuisance variation while preserving the geometric and semantic cues of true change.

The LEVIR-CD paper frames this challenge in terms of spatial-temporal relationships: a model can benefit from comparing pixels not only at the same spatial coordinate, but also across neighboring or distant positions and across both acquisition times (Chen & Shi, 2020). This observation is central to the motivation of the present work. The final model does not treat the two images as unrelated channels; instead, it learns a shared representation and then performs attention over the joint temporal token set.

2.2 CNNs for Dense Prediction

Convolutional neural networks are natural candidates for satellite image analysis because they learn translation-equivariant filters and progressively combine local evidence. For dense prediction, a convolutional model maps an input tensor to an output tensor with spatial correspondence. In the simplest case, a stack of padded convolutions preserves spatial resolution and predicts a binary logit for each pixel. Such a model is easy to train and interpret, but it has limited context aggregation. Even with dilated convolutions, the receptive field remains constrained by the number of layers and dilation pattern.

The simple CNN baseline in this work is intentionally shallow. It concatenates the two RGB images into a six-channel tensor and applies a sequence of convolutional blocks with dilation rates selected to increase receptive field while avoiding repeated sparse sampling patterns. Since the model contains no pooling and no decoder, it has no risk of resolution mismatch. However, it also lacks the explicit multi-scale representation that is typically useful when buildings vary in size and shape.

2.3 U-Net for Semantic Segmentation

The second architecture adopts an encoder-decoder structure with skip connections. The encoder progressively reduces spatial resolution while increasing channel depth, allowing the network to aggregate broader context. The decoder then restores the spatial resolution by learned upsampling and combines the upsampled representation with skip features from the encoder. This structure is well suited to binary change segmentation because it can combine local edge detail with broader object-level

context.

In this project, the U-Net is implemented as an early-fusion model. The two images are concatenated before the first convolution, and the network must learn temporal comparison implicitly. This design is straightforward and strong: if sufficient data are available, the first layers can learn low-level differences, while deeper layers can learn object-level change patterns. However, the architecture does not impose any structural separation between the two timestamps. The model sees a six-channel image, not two temporally distinct observations.

2.4 Siamese Networks for Bitemporal Imagery

A Siamese architecture introduces a more appropriate inductive bias for bitemporal inputs. The two images are processed by the same encoder, meaning that the feature extractor is shared across time. If the images contain the same type of content, this shared representation encourages comparable feature spaces for t_1 and t_2. Given images A and B, the encoder produces multi-resolution feature sets

$$\mathcal{F}_A = E(A), \qquad \mathcal{F}_B = E(B), \tag{1}$$

where E denotes the shared encoder. Feature fusion can then use both the two representations and their absolute difference. This structure explicitly separates feature extraction from temporal comparison.

The Siamese Temporal Attention U-Net used in this paper builds on this idea. The shared encoder produces corresponding feature maps at multiple resolutions. At the bottleneck, where spatial resolution is reduced and semantic abstraction is high, a global temporal attention block allows each encoded position from either timestamp to attend to all encoded positions from both timestamps. The decoder then uses temporally fused skip features to reconstruct a full-resolution change mask.

2.5 Attention Mechanisms for Temporal Feature Comparison

Attention mechanisms compute adaptive weighted combinations of features. In the context of bitemporal change detection, the key benefit is that correspondence does not need to be restricted to the same pixel coordinate. If a building boundary is slightly shifted between images, or if a local region is affected by shadows, a temporal attention module can compare a feature with a broader set of candidate features across both time and space. The LEVIR-CD paper motivates attention as a way to capture spatial-temporal dependencies that help mitigate illumination changes and misregistration artifacts (Chen & Shi, 2020).

In this work, attention is applied at the bottleneck of the Siamese encoder. This choice keeps the token count computationally manageable while still allowing global interactions at a semantically meaningful level. Unlike early fusion, which leaves temporal comparison entirely implicit, the attention formulation makes the temporal relationship explicit through query, key, and value projections applied to the concatenated temporal feature set.

3 Dataset and Exploratory Data Analysis

3.1 LEVIR-CD Dataset

The experiments are conducted on LEVIR-CD, a remote sensing change detection dataset introduced by Chen and Shi (Chen & Shi, 2020). The dataset contains 637 bitemporal image pairs of size 1024×1024, acquired from Google Earth, with more than 31,000 labeled building change instances. Each sample consists of an image before change, an image after change, and a binary mask indicating changed building pixels. The dataset is appropriate for evaluating deep learning models because it contains a substantially larger number of labeled instances than earlier small-scale change detection benchmarks described in the LEVIR-CD paper (Chen & Shi, 2020).

The official images are split into training, validation, and test partitions. In our preprocessing pipeline, each 1024×1024 image is divided into non-overlapping 256×256 patches. This patching strategy preserves the original image content while reducing memory requirements and increasing the number of training examples. The resulting split statistics are reported in Table 1. The train split contains 445 image pairs and 7120 patches, the validation split contains 64 image pairs and 1024 patches, and the test split contains 128 image pairs and 2048 patches.

Table 1: Dataset split statistics after non-overlapping 256×256 patch extraction. The change ratio is the fraction of positive pixels in the corresponding split.

Split	Image pairs	Patches	Changed pixels	Change ratio	Empty patch fraction
Train	445	7120	21,412,971	0.0459	0.5552
Validation	64	1024	2,816,268	0.0420	0.5742
Test	128	2048	6,837,404	0.0509	0.5435

3.2 Image Pairs and Binary Change Masks

Each sample is represented as a triplet (A_i, B_i, Y_i), where $A_i \in \mathbb{R}^{3\times H\times W}$ is the pre-event RGB image, $B_i \in \mathbb{R}^{3\times H\times W}$ is the post-event RGB image, and $Y_i \in \{0,1\}^{1\times H\times W}$ is the binary target mask. Figure 1 shows a representative image pair and its ground truth mask. This form of supervision makes the task a dense binary segmentation problem rather than image-level classification.

(a) Image at t_1. **(b)** Image at t_2. **(c)** Change mask.

Figure 1: Example bitemporal LEVIR-CD patch used in the experimental pipeline. The model receives the two RGB images and predicts the binary building-change mask. Additional qualitative outputs are reported in Appendix C.

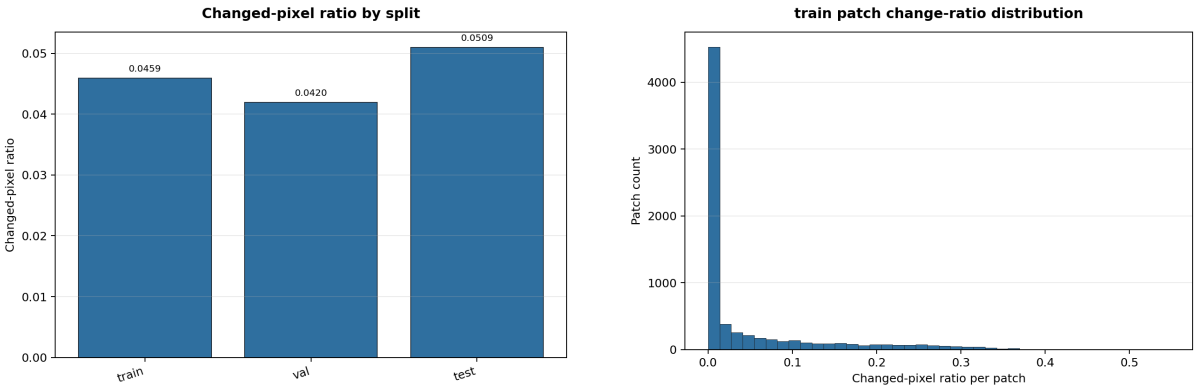
3.3 Patch Extraction and Preprocessing

All images and masks are resized or cropped to a common 256×256 patch size. Since the dataset images are 1024×1024, non-overlapping patch extraction yields sixteen patches per original image pair. RGB values are converted to floating-point tensors and normalized using the shared dataset statistics saved during preprocessing. Masks are converted to binary tensors with shape $1 \times H \times W$. During training, random spatial augmentations are applied jointly to the two images and the mask, preserving geometric consistency across the bitemporal pair.

The use of a common preprocessing pipeline is important for experimental validity. The three models differ only in architecture and hyperparameters, not in input resolution, normalization, data split, or mask encoding. This avoids a common confound in change detection comparisons: a model may appear stronger simply because it receives different preprocessing or a different effective input distribution.

3.4 Class Imbalance

Building changes are sparse. In the train split, changed pixels account for only approximately 4.59% of all pixels, while the median patch-level change ratio is zero. More than half of the extracted patches contain no changed pixels at all. This imbalance is visualized in Figure 2. The sparsity of the positive class has two implications. First, accuracy alone is not a meaningful evaluation metric, because a model can obtain high accuracy by predicting most pixels as unchanged. Second, the loss function must counteract the imbalance to prevent the model from converging to a trivial background-dominated solution.



(a) Positive pixel ratio per split.

(b) Training patch change-ratio distribution.

Figure 2: Exploratory analysis of class imbalance after patch extraction. The majority of pixels belong to the unchanged class, and many patches contain no changed pixels. This motivates the weighted BCE–Dice training objective used in Section 4.2.

3.5 Train, Validation, and Test Protocol

The training set is used to optimize network weights. The validation set is used for early stopping and hyperparameter selection. The test set is held out until final evaluation. All reported final metrics in Section 5 correspond to the test split. This separation is especially important because hyperparameter optimization was performed for the U-Net and Siamese Temporal Attention U-Net. The reported test results therefore measure generalization after model selection rather than validation performance alone.

4 Methodology

4.1 Shared Preprocessing Pipeline

Let $\mathcal{D} = \{(A_i, B_i, Y_i)\}_{i=1}^{N}$ denote the patch-level dataset. Each image pair is transformed into normalized tensors

$$\tilde{A}_i = \frac{A_i - \mu}{\sigma}, \qquad \tilde{B}_i = \frac{B_i - \mu}{\sigma}, \tag{2}$$

where μ and σ are channel-wise statistics estimated from the training images. The same normalization is applied to both timestamps. The target mask is kept binary. The networks output raw logits $Z_i \in \mathbb{R}^{1\times H\times W}$, which are converted to probabilities by the sigmoid function

$$P_i = \sigma(Z_i) = \frac{1}{1 + \exp(-Z_i)}. \tag{3}$$

During evaluation, the predicted binary mask is obtained by thresholding P_i at $\tau = 0.5$ unless otherwise specified.

4.2 Loss Function: Weighted BCE and Dice Loss

The training objective combines weighted binary cross-entropy with Dice loss. The weighted BCE term addresses class imbalance by assigning a higher cost to positive pixels. For a pixel target $y \in \{0, 1\}$ and predicted probability p, the weighted BCE can be written as

$$\mathcal{L}_{\mathrm{BCE}}(p, y) = -\alpha y \log(p) - (1 - y) \log(1 - p), \tag{4}$$

where α is the positive-class weight. In implementation, the loss is computed directly from logits using a numerically stable BCE-with-logits formulation.

The Dice loss complements BCE by directly optimizing overlap between predicted and target masks. For a batch of predicted probabilities P and targets Y, the soft Dice coefficient is

$$\mathrm{Dice}(P, Y) = \frac{2\sum_j P_j Y_j + s}{\sum_j P_j + \sum_j Y_j + s}, \tag{5}$$

where s is a smoothing constant. The Dice loss is

$$\mathcal{L}_{\mathrm{Dice}} = 1 - \mathrm{Dice}(P, Y). \tag{6}$$

The final loss is a convex combination

$$\mathcal{L} = \lambda_{\mathrm{BCE}}\mathcal{L}_{\mathrm{BCE}} + (1 - \lambda_{\mathrm{BCE}})\mathcal{L}_{\mathrm{Dice}}. \tag{7}$$

The coefficient λ_{BCE} and the positive-weight scaling factor are model-specific hyperparameters selected during optimization.

4.3 Evaluation Metrics

The central metric is F1-score, because it balances precision and recall in the presence of class imbalance. Let TP, FP, FN, and TN denote true positives, false positives, false negatives, and true negatives over all pixels in the test split. Precision, recall, F1-score,

and intersection-over-union (IoU) are computed as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{8}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{9}$$

$$\text{F1} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}, \tag{10}$$

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}. \tag{11}$$

Dice coefficient is equivalent to the F1-score under binary segmentation when computed from the same confusion counts. Accuracy is reported for completeness, but it is not used as the primary criterion because unchanged pixels dominate the dataset.

4.4 Simple CNN Baseline

The baseline model receives the concatenated tensor $[\tilde{A}; \tilde{B}] \in \mathbb{R}^{6\times H\times W}$ and processes it with a shallow sequence of padded convolutional blocks. Each block consists of a convolution, batch normalization, and ReLU activation. The dilation pattern is $1, 1, 2, 3, 1$, increasing the receptive field without downsampling. The final 1×1 convolution maps the last feature map to a single logit channel. Because the model operates at full resolution throughout, it does not use pooling, upsampling, or skip connections.

This architecture is intentionally limited. It provides a lower bound for the comparison and tests whether local texture and color differences are sufficient for building change detection. The results in Section 5 show that this is not the case: the model achieves high recall but much lower precision than the segmentation networks, indicating a tendency to over-detect change regions.

4.5 Early-Fusion U-Net

The U-Net also receives the concatenated six-channel tensor, but it introduces a four-stage encoder, a bottleneck, and a symmetric decoder. The encoder channel dimensions are 32, 64, 128, and 256, followed by a 512-channel bottleneck. Each encoder block applies two convolutional layers with batch normalization and ReLU activations. Max-pooling reduces spatial resolution, and transposed convolutions restore it in the decoder. Skip connections concatenate encoder features with decoder features at matching resolutions.

Formally, if S_l denotes the encoder feature at level l and D_{l+1} denotes the decoder feature from the coarser level, the decoder update can be described as

$$D_l = g_l\left([S_l; \text{Up}(D_{l+1})]\right), \tag{12}$$

where g_l is a double-convolution block and $[\cdot;\cdot]$ denotes channel-wise concatenation. This design allows the decoder to recover fine spatial detail while using semantic context from the bottleneck.

4.6 Siamese Temporal Attention U-Net

The final model separates temporal feature extraction from temporal comparison. The two RGB images are passed through the same encoder. In implementation, they are concatenated along the batch dimension and processed in a single encoder pass, ensuring shared weights and stable batch-normalization statistics. The resulting feature dictionaries are then split back into features for time t_1 and t_2.

At each skip level, features are fused using the two temporal representations and their absolute difference:

$$S_l^{\text{fuse}} = \phi_l \left([S_l^A; S_l^B; |S_l^A - S_l^B|] \right), \tag{13}$$

where ϕ_l is a 1×1 convolution followed by batch normalization and ReLU activation. This fusion rule makes the local temporal difference explicit while still preserving the original features from both timestamps.

The main architectural addition is the temporal attention block at the bottleneck. Let

$$F_A, F_B \in \mathbb{R}^{C \times H' \times W'} \tag{14}$$

denote bottleneck feature maps for the two timestamps. These are projected into query, key, and value tensors:

$$Q = W_Q F, \qquad K = W_K F, \qquad V = W_V F, \tag{15}$$

where F denotes the temporal concatenation of F_A and F_B over the token dimension. After flattening the spatial dimensions, the token set has length

$$T = 2H'W', \tag{16}$$

because it contains every spatial position from both timestamps. The attention matrix is computed as

$$\mathcal{A} = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}} \right), \tag{17}$$

where d is the reduced query-key dimension. The attended features are

$$O = \mathcal{A}V. \tag{18}$$

Finally, a residual projection is applied:

$$\hat{F} = F + \gamma W_O O, \tag{19}$$

where γ is a learnable scalar initialized at zero. This initialization makes the attention block behave initially like an identity mapping, reducing optimization instability. The attended temporal features are split into $\hat{F}_A$ and $\hat{F}_B$, fused using Eq. (13), and decoded with the same skip-connection structure as the U-Net.

Figure 3 summarizes the three model families. The detailed implementation of the temporal attention block is shown in Appendix A.2.

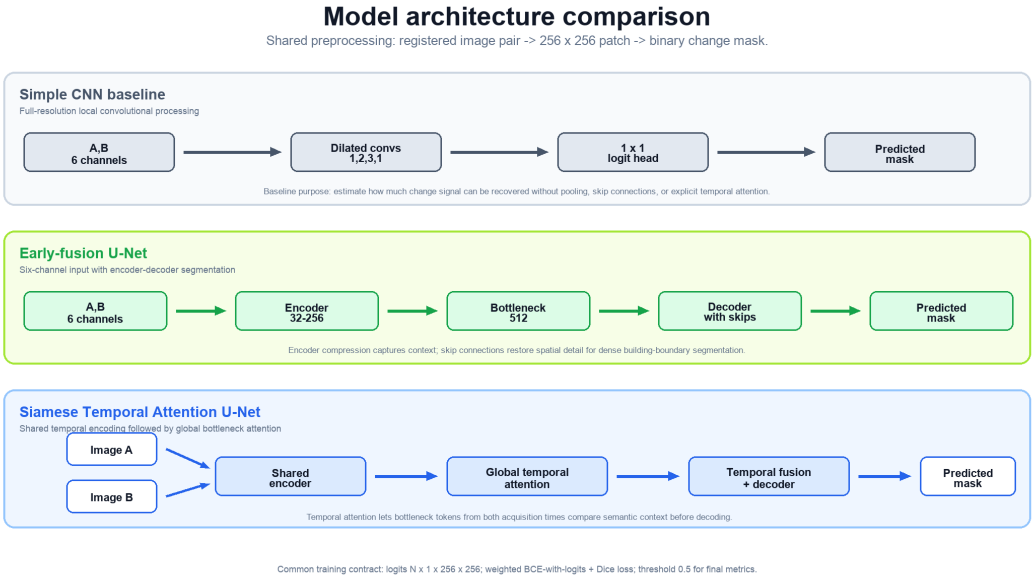


Figure 3: Conceptual architecture comparison. The simple CNN performs full-resolution local processing, the U-Net introduces early fusion and encoder-decoder segmentation, and the Siamese Temporal Attention U-Net encodes both timestamps with shared weights before applying temporal attention at the bottleneck.

4.7 Hyperparameter Optimization with Optuna

The baseline CNN uses a fixed conservative configuration, while U-Net and Siamese Temporal Attention U-Net use hyperparameters selected through Optuna experiments. The optimized parameters include learning rate, weight decay, BCE–Dice mixing coefficient, positive-weight scale, early stopping rule, and gradient clipping. The optimization objective is validation F1-score. Since F1-score is sensitive to both precision and recall, it is better aligned with the final evaluation goal than validation loss alone.

The final hyperparameters are reported in Table 2. The U-Net is trained for at most 80 epochs, reflecting the budget used during its optimization. The Siamese Temporal Attention model is trained for at most 100 epochs because its best optimized configuration reached the best validation region later in training. Both segmentation models use early stopping on validation F1-score with patience 12 and minimum improvement 5×10^{-4}.

Table 2: Final training hyperparameters. The BCE weight is λ_{BCE} in Eq. (7). The Dice weight is $1 - \lambda_{\mathrm{BCE}}$.

Model	Epoch cap	LR	Weight decay	BCE weight	Dice weight	Pos. scale
Simple CNN	150	1.00×10^{-3}	1.00×10^{-4}	0.5000	0.5000	1.0000
U-Net	80	2.93×10^{-4}	4.56×10^{-5}	0.3003	0.6997	0.3467
Siamese Temporal Attention	100	3.58×10^{-4}	5.06×10^{-5}	0.3016	0.6984	0.2104

4.8 Training Setup and Early Stopping

All models are trained with AdamW, cosine learning-rate annealing, batch size 8, and automatic mixed precision when CUDA is available. The threshold for validation and test metrics is fixed at 0.5. The simple CNN uses early stopping on validation loss with patience 35, while the two segmentation models use early stopping on validation F1-score with patience 12. This difference reflects the role of the CNN as a broad baseline and the

role of the two optimized models as final candidates selected for F1-score.

The training procedure is summarized in Appendix A.3. Each epoch computes train metrics and validation metrics, updates the learning-rate scheduler, saves the best checkpoint according to validation F1-score, and applies early stopping according to the model-specific monitor. At the end of training, the best checkpoint is restored and evaluated on the test split.

5 Results

5.1 Overall Model Comparison

The main quantitative comparison is reported in Table 3. This table is the central result of the experimental study: it compares the three final models on the same held-out test split, using the same preprocessing pipeline, loss formulation, decision threshold, and evaluation metrics. The models are ordered according to the architectural progression followed in the project: Simple CNN is the full-resolution convolutional baseline, U-Net is the early-fusion encoder-decoder model, and Siamese TA U-Net denotes the Siamese Temporal Attention U-Net.

Table 3: Main test-set comparison between the three final models. ΔF1 is computed relative to the preceding model in the architectural progression.

Model	Test F1	Test IoU	Precision	Recall	Test loss	ΔF1
Simple CNN	0.7093	0.5496	0.5895	0.8903	0.5404	–
U-Net	0.8840	0.7922	0.8787	0.8895	0.1681	+0.1747
Siamese TA U-Net	0.9019	0.8214	0.9000	0.9039	0.1189	+0.0179

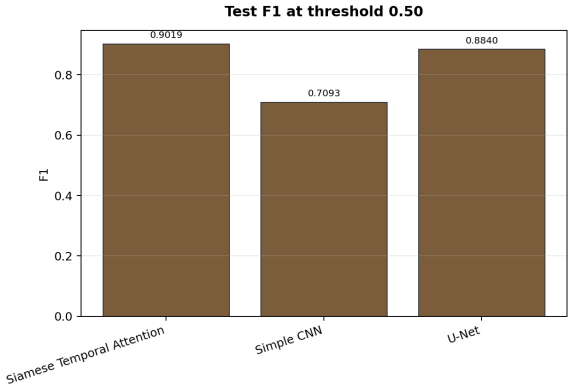
The results show a clear performance hierarchy. The simple CNN obtains a test F1-score of 0.7093, showing that local full-resolution convolutions can detect part of the change signal but are not sufficient for high-quality dense segmentation. The U-Net improves test F1-score by 0.1747, confirming the importance of encoder-decoder context aggregation and skip-connected spatial recovery. The Siamese Temporal Attention U-Net obtains the best result, with a test F1-score of 0.9019 and a test IoU of 0.8214. Its improvement of 0.0179 F1-score over U-Net is smaller than the jump from CNN to U-Net, but it is meaningful because both models already share the same segmentation backbone family and differ mainly in their explicit temporal comparison mechanism.

5.2 Final Training Configuration

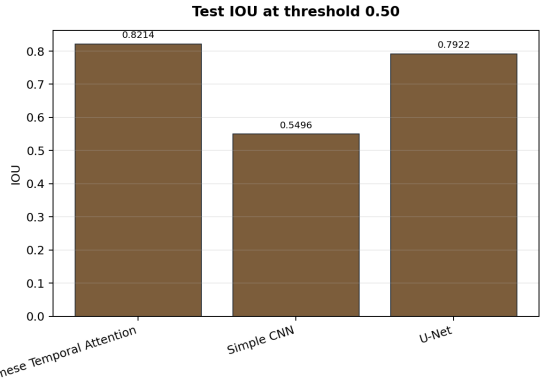
The final run was executed on a CUDA-enabled HPC node with an NVIDIA A100 MIG GPU. The three models were trained under the shared dataset pipeline described in Section 3. The simple CNN stopped after 139 epochs, with best validation F1-score at epoch 120. The U-Net stopped after 51 epochs, with best validation F1-score at epoch 39. The Siamese Temporal Attention U-Net stopped after 69 epochs, with best validation F1-score at epoch 63.

The progression is also shown graphically in Figure 4. The most important comparison is between U-Net and Siamese Temporal Attention U-Net. Both models use encoder-decoder segmentation and comparable training protocols; the main difference is the explicit temporal encoding and attention mechanism. The improvement from 0.8840 to

0.9019 test F1-score indicates that temporal attention provides a measurable advantage beyond the U-Net backbone.



(a) Test F1-score.

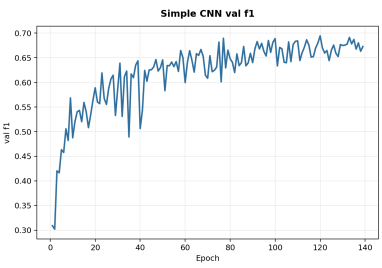


(b) Test IoU.

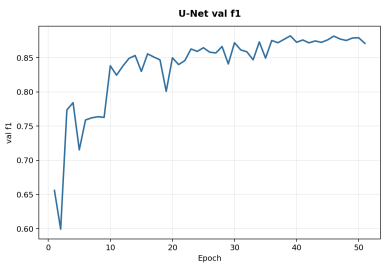
Figure 4: Final quantitative comparison on the held-out test set. The Siamese Temporal Attention U-Net obtains the highest F1-score and IoU, while the simple CNN baseline remains substantially lower.

5.3 Training Curves

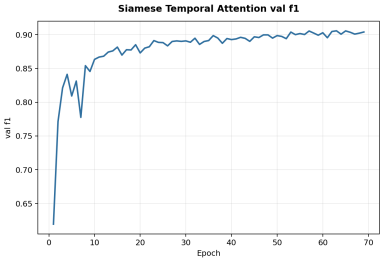
Figure 5 reports validation F1 curves for the three models. The simple CNN improves slowly and remains unstable, reflecting its limited ability to aggregate object-level context. The U-Net rapidly reaches high validation performance and then saturates, triggering early stopping. The Siamese Temporal Attention U-Net continues improving beyond the early U-Net plateau and reaches the best validation F1-score among all models.



(a) Simple CNN.



(b) U-Net.

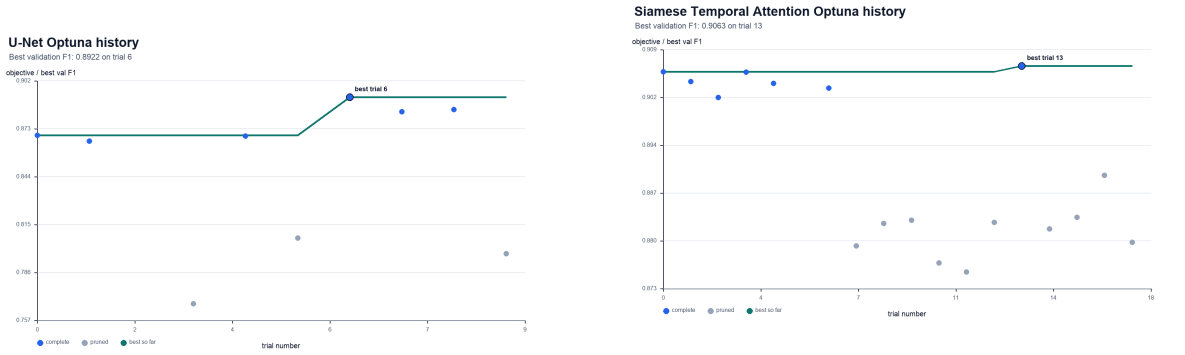


(c) Siamese Temporal Attention.

Figure 5: Validation F1-score curves for the three final models. The segmentation models converge to substantially higher F1-scores than the shallow CNN baseline. Additional training curves are included in Appendix B.

5.4 Hyperparameter Analysis

The hyperparameter optimization traces for U-Net and Siamese Temporal Attention U-Net are shown in Figure 6. The U-Net search identifies a strong configuration by trial 6, while the Siamese search reaches its final best configuration at trial 13. The parameter-importance visualization in Appendix B should be interpreted as a correlation-based analysis rather than a causal attribution method. It is useful for summarizing which hyperparameters varied together with validation performance in the finite search space, but it does not prove that a parameter alone caused the observed improvement.



(a) U-Net optimization history.

(b) Siamese Temporal Attention optimization history.

Figure 6: Static Optuna optimization histories for the two optimized models. The objective is validation F1-score.

5.5 Threshold Sensitivity

Although the final metrics are reported at threshold $\tau = 0.5$, threshold-sensitivity plots were generated for each model. These plots quantify how precision, recall, and F1-score vary when the decision threshold changes. This analysis is important because class imbalance can make a model appear strong at one threshold and weak at another. In this experiment, the selected threshold provides a consistent basis for comparison across models. The threshold plots are included in Appendix B.

5.6 Qualitative Prediction Examples

Figure 7 shows a qualitative comparison on a representative test patch. The simple CNN tends to produce noisier masks, while the U-Net produces more coherent regions. The Siamese Temporal Attention model produces the closest visual alignment with the ground truth in the selected sample. Additional overlays and error maps are shown in Appendix C.

(a) Ground truth. **(b)** Simple CNN. **(c)** U-Net. **(d)** Siamese attention.

Figure 7: Qualitative mask comparison on a held-out sample. The final model more closely matches the ground truth mask while suppressing some false positives visible in the simpler models.

6 Discussion

6.1 Baseline CNN Behavior

The simple CNN baseline is useful because it isolates the effect of local convolutional processing. Its test recall is high, 0.8903, but its precision is only 0.5895. This indicates that the model detects many changed pixels but also produces a large number of false positives. The result is consistent with the architecture: without an encoder-decoder structure, the model has limited ability to reason about object extent and context. Dilated convolutions help enlarge the receptive field, but they do not provide the hierarchical representation needed to distinguish true building changes from texture-level differences.

The baseline should therefore not be interpreted as a failed model. Rather, it establishes that the task cannot be solved reliably by shallow local processing alone. It also confirms that the experimental protocol is sensitive enough to separate weak and strong architectures: the difference between 0.7093 and 0.8840 test F1-score is large and meaningful.

6.2 U-Net as a Strong Segmentation Baseline

The U-Net produces a major improvement over the baseline. The encoder aggregates context, while the decoder and skip connections preserve spatial detail. This is particularly important for building change detection, where object boundaries and small structures must be recovered accurately. The U-Net reaches a test F1-score of 0.8840 and a test IoU of 0.7922, demonstrating that early fusion plus multi-scale segmentation is already a strong approach.

However, early fusion has a limitation: temporal comparison is not structurally enforced. The model receives a six-channel tensor and must learn from data how the first three channels relate to the last three. This is flexible, but it does not explicitly encode the symmetry of bitemporal comparison. The Siamese model addresses this by using a shared encoder and feature-level temporal fusion.

6.3 Why Temporal Attention Improves Change Detection

The Siamese Temporal Attention U-Net improves test F1-score by 0.0179 over U-Net. This gain is not merely a result of adding parameters; it reflects a better alignment between the architecture and the task. Change detection is fundamentally comparative. The model must decide whether a structure in A corresponds to a structure in B, and whether any difference is semantically meaningful. The shared encoder produces comparable representations for both timestamps, while the temporal attention block allows each bottleneck token to access evidence from all encoded positions in both images.

The attention formulation in Eq. (17) is particularly relevant in the presence of small spatial misalignments. A feature at one location in A can attend to nearby or distant features in B, not only to the feature at the same coordinate. This does not eliminate the need for accurate co-registration, but it gives the network a mechanism for learning robust temporal correspondences. The residual initialization in Eq. (19) also makes the attention block conservative at the beginning of training, allowing the network to learn a useful attention correction rather than forcing a disruptive transformation from the first iteration.

6.4 Error Analysis

The qualitative maps reveal that most errors occur near object boundaries and in regions where visual evidence is ambiguous. The simple CNN often produces scattered false positives because local differences are not always semantic changes. The U-Net reduces

this noise but may still miss small changed structures or produce boundary inaccuracies. The Siamese Temporal Attention model reduces some of these errors, but it is not perfect. It can still fail when the changed region is extremely small, when the contrast between building and background is weak, or when the annotation boundary is sharper than the visual evidence.

This behavior is expected for pixel-level remote sensing segmentation. Labels are binary and precise, while the visual signal may be gradual or uncertain. Therefore, small boundary errors can affect IoU and F1-score even when the prediction is qualitatively reasonable. The error map in Appendix C illustrates these residual failure modes.

6.5 Computational Trade-Offs

The Siamese Temporal Attention U-Net has more parameters than the U-Net because it uses a shared encoder, temporal attention projections, and feature-fusion blocks. The attention module also introduces quadratic cost in the number of bottleneck tokens. Since attention is applied after four pooling stages, the bottleneck resolution for 256×256 input patches is small enough for training on the available GPU. Applying global attention at higher resolutions would be substantially more expensive. The chosen design therefore represents a practical trade-off: temporal attention is applied where the semantic abstraction is high and the token count is manageable.

6.6 Future Improvement: Extension to SAR Imagery

A natural extension of this work is the adaptation of the same bitemporal change-detection pipeline to Synthetic Aperture Radar (SAR) imagery. SAR sensors are active microwave instruments: instead of measuring reflected sunlight like optical sensors, they emit a radar signal and measure the backscattered response from the Earth's surface. This makes SAR particularly attractive for operational monitoring because acquisitions can be performed independently of daylight conditions and are far less affected by cloud cover than optical imagery. For applications such as disaster response, flood mapping, infrastructure monitoring, and large-scale territorial surveillance, this property can be decisive.

From a methodological perspective, switching to SAR would not require replacing the full learning framework. The present project already separates preprocessing, dataset handling, model definitions, losses, metrics, and training logic. Therefore, the core architectures could remain comparable while the input representation and preprocessing steps are adapted. In the optical LEVIR-CD setting, each timestamp is represented by three RGB channels. In a SAR setting, a common representation would instead use calibrated backscatter channels such as VV and VH polarization. A bitemporal early-fusion input could therefore change from six optical channels to four SAR channels, while a Siamese model would process the two SAR acquisitions through the same shared encoder.

The main technical changes would concern data preparation. SAR images usually require radiometric calibration, conversion to a logarithmic decibel scale, robust clipping of extreme values, and normalization adapted to the statistics of radar backscatter. They also contain speckle noise, a granular interference pattern typical of coherent radar imaging, which may require filtering or data augmentation strategies that differ from optical imagery. Importantly, this extension does not necessarily require working with raw SAR signals or introducing Fourier-domain processing. For a first deep learning implementation, it would be sufficient to use preprocessed SAR products and treat them as multi-channel image tensors, while preserving the same supervised segmentation

framework.

The expected benefit of this extension is not merely a change of sensor modality. It would move the project from optical building change detection toward a more operational Earth observation setting. A SAR-based version could evaluate whether temporal attention remains useful when changes are expressed through radar backscatter rather than visible texture and color. It could also support tasks where SAR is especially strong, such as flood extent segmentation, post-event change detection, or monitoring of areas where optical imagery is frequently limited by weather conditions. In this sense, SAR imagery represents a promising future direction for testing the generality and practical value of the proposed bitemporal architecture.

7 Conclusion

This paper presented a complete experimental study for building change detection on LEVIR-CD. Three models were compared under a shared preprocessing, training, and evaluation pipeline: a simple full-resolution CNN, an early-fusion U-Net, and a Siamese Temporal Attention U-Net. The results show a clear progression in performance. The CNN baseline reaches 0.7093 test F1-score, demonstrating that local convolutional processing alone is insufficient. The U-Net reaches 0.8840, confirming the importance of multi-scale encoder-decoder segmentation. The Siamese Temporal Attention U-Net reaches 0.9019, showing that explicit temporal representation and global bottleneck attention improve change detection beyond early fusion.

The study supports the conclusion that architecture should reflect the structure of the data. Bitemporal change detection is not simply semantic segmentation on a six-channel image; it is a comparison between two temporally separated observations. A shared encoder and temporal attention mechanism provide a more suitable inductive bias for this comparison. Future work may explore larger attention capacity, multi-threshold calibration, uncertainty estimation, and additional qualitative analysis across different urban densities and building sizes. A particularly relevant direction is the extension from optical RGB imagery to SAR imagery, where weather-independent and illumination-independent acquisitions could make the same methodological framework more suitable for operational monitoring scenarios.

A Additional Technical Details

This appendix collects implementation details that support the main text but would interrupt the flow of the methodology section. The main experimental design references this appendix in Sections 4 and 5.

A.1 Architecture Details

The simple CNN baseline consists of five convolutional blocks followed by a 1×1 logit head. The U-Net uses encoder channels 32, 64, 128, 256 and a 512-channel bottleneck. The Siamese Temporal Attention U-Net uses the same channel schedule for the shared encoder, applies attention at the 512-channel bottleneck, and fuses skip features at all levels through 1×1 convolutional fusion blocks. The parameter counts are reported in Table 4.

Table 4: Trainable parameter counts for the three final models.

Model	Trainable parameters
Simple CNN	59,745
U-Net	7,763,905
Siamese Temporal Attention U-Net	9,403,554

A.2 Key Temporal Attention Implementation

The following code snapshot shows the core of the temporal attention block. The feature maps from both timestamps are concatenated along the batch dimension, projected to query, key, and value tensors, reshaped into a token sequence of length $2H'W'$, and processed by scaled dot-product attention.

Listing 1: Core temporal attention computation used in the Siamese Temporal Attention U-Net.

```
query = self.query(x).reshape(2, batch, -1, height * width).permute(1,
    0, 3, 2)
key = self.key(x).reshape(2, batch, -1, height * width).permute(1, 0,
    3, 2)
value = self.value(x).reshape(2, batch, channels, height * width).
    permute(1, 0, 3, 2)

query = query.reshape(batch, 2 * height * width, -1)
key = key.reshape(batch, 2 * height * width, -1)
value = value.reshape(batch, 2 * height * width, channels)

attention = torch.softmax((query @ key.transpose(1, 2)) / math.sqrt(key
    .shape[-1]), dim=-1)
out = attention @ value
out = out.reshape(batch, 2, height * width, channels).permute(1, 0, 3,
    2)
out = out.reshape(2 * batch, channels, height, width)
out = x + self.gamma * self.proj(out)
```


A.3 Training Loop Snapshot

Listing 2 shows the essential training logic. The checkpoint is updated when validation F1-score improves, while early stopping monitors the metric specified by each model configuration.

Listing 2: Simplified training loop logic.

```
for epoch in range(1, epochs + 1):
    train_metrics = run_epoch(model, loaders["train"], criterion,
       device, train=True)
    val_metrics = run_epoch(model, loaders["val"], criterion, device,
       train=False)
    scheduler.step()

    if val_metrics["f1"] > best_f1:
        best_f1 = val_metrics["f1"]
        best_epoch = epoch
        torch.save({"model_state": model.state_dict(),
                    "best_f1": best_f1,
                    "epoch": epoch}, best_path)

    monitor_value = {"val_loss": val_metrics["loss"],
                     "val_f1": val_metrics["f1"],
                     "val_iou": val_metrics["iou"]}[early_stop_metric]
    if early_stopping.step(monitor_value):
        break
```

A.4 Loss Implementation Snapshot

Listing 3 shows the BCE–Dice composition used throughout the experiments. The implementation checks that logits and targets have identical shapes, preventing silent broadcasting errors.

Listing 3: Weighted BCE–Dice loss implementation.

```
class BCEDiceLoss(nn.Module):
    def __init__(self, pos_weight=None, bce_weight=0.5):
        super().__init__()
        self.register_buffer("pos_weight", torch.tensor([float(
           pos_weight)]))
        self.bce = nn.BCEWithLogitsLoss(pos_weight=self.pos_weight)
        self.dice = DiceLoss()
        self.bce_weight = float(bce_weight)

    def forward(self, logits, targets):
        assert_same_shape(logits, targets)
        bce = self.bce(logits, targets.float())
        dice = self.dice(logits, targets)
        return self.bce_weight * bce + (1.0 - self.bce_weight) * dice
```

B Additional Figures

Figure 8 reports the correlation-based hyperparameter association plot for the Siamese Temporal Attention search. Figure 9 reports the threshold-sensitivity curves generated by the project pipeline.

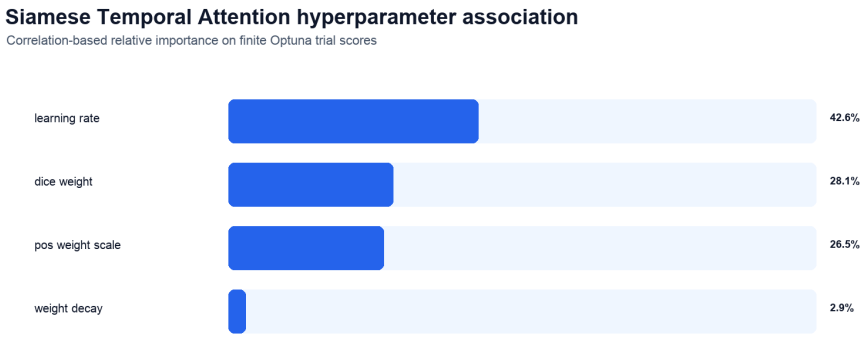


Figure 8: Correlation-based hyperparameter association plot for the Siamese Temporal Attention optimization. This figure complements the optimization histories in Figure 6.

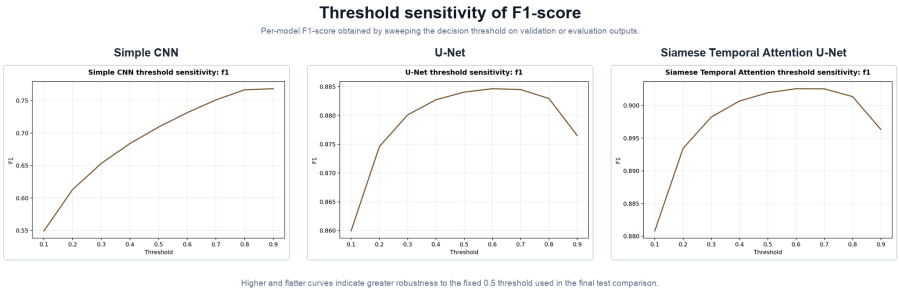


Figure 9: Threshold-sensitivity summary for the three final models. The curves show how F1-score varies when the decision threshold is swept, complementing the fixed-threshold results reported in Table 3.

C Qualitative Case Studies

The qualitative outputs in Figure 10 complement the main prediction comparison in Figure 7. Error maps are useful because they separate false positives and false negatives, making it easier to identify whether a model tends to over-detect or under-detect changes.

(a) Reference image at t_2. **(b)** Predicted mask. **(c)** Error map.

Figure 10: Qualitative case study for the Siamese Temporal Attention U-Net. The error map highlights residual boundary and small-object errors.

D Reproducibility Notes

The final project directory contains a compact submission folder with source code, configuration files, training scripts, SLURM scripts, and figure-generation utilities. The final training command loads the fixed hyperparameter configuration and automatically selects CUDA when available. The figure-generation script loads the trained checkpoints and produces one image per plot or visual representation, avoiding multi-panel files when individual figures are easier to include in reports.

Listing 4: Final training and figure-generation commands on the HPC environment.

```
cd ~/levir_project/levir_final/submission
sbatch slurm/run_training.slurm

# After training finishes:
RUN_DIR=results/final_run_20260715_120228 sbatch slurm/generate_figures
    .slurm
```


References

Chen, H., & Shi, Z. (2020). A spatial-temporal attention-based method and a new dataset for remote sensing image change detection. *Remote Sensing*, *12*(10), 1662. https://doi.org/10.3390/rs12101662