earthquake_CNN

Release 0.1

Paolo Bonfini

Jul 01, 2025

CONTENTS:

CHAPTER

ONE

PROJECT OVERVIEW

This project focuses on simulating and predicting the effects of earthquakes on synthetic buildings composed of bays arranged in a 3D grid of size **R × C × K**, where **R** is the number of floors, and **C** and **K** are the number of bays along the building's width and depth, respectively. Each building configuration is subjected to simulated seismic events, and the resulting structural stress is analyzed using **finite element analysis (FEA)**.

This synthetic dataset is used to create a machine learning model that can predict post-earthquake stress distributions based on pre-earthquake geometry and loading.

1.1 Objective

The primary objective of this project is to develop a **convolutional neural network (CNN)** model capable of predicting the stress experienced by each bay in the building following an earthquake. The model takes as input the building's structural characteristics—such as its geometry and bay layout—along with the seismic input parameters, including the earthquake's peak ground acceleration (PGA) and frequency.

By learning from the synthetic dataset generated through FEA simulations, the CNN aims to estimate the resulting stress distribution across the building bays with high spatial resolution. This approach can potentially enable real-time assessment of structural vulnerability without the need for computationally expensive simulations.

CHAPTER

TWO

DATA AND PREPROCESSING

2.1 Images

Our data **images** consist of a collection of 3,421 synthetic buildings rendered from **four viewpoints**, corresponding to the cardinal directions around the structure, i.e., front, back, left, and right view. For each building, we therefore have:

- **4 input images** showing the structure from four cardinal directions
- **4 output images** visualizing the resulting stress after an earthquake, also from the same four directions

These views are captured **both before and after** the simulated earthquake:

- The **pre-quake images** represent the building geometry
- The **post-quake images** display stress distributions computed via finite element analysis (FEA)

The post-quake images use **color-coded bays** to indicate the level of stress — higher stress regions are shown in warmer colors (e.g., red), and lower stress in cooler tones (e.g., blue or green).

2.1.1 Example pre-/post-Earthquake

Below is an example of one building's input and its corresponding output after the application of the earthquake, both from the front view:

Note that our analysis focuses only on a **subset** of the total bays within each building (in the examples above, the third column of bays). In this representation, the finite elements are shown as a mesh that subdivides the central bays. The mesh resolution and bay layout are consistent across all buildings, ensuring that stress patterns can be compared between different designs. However, they may correspond to different physical sizes. This size and layout information is stored in accompanying metadata files provided alongside the images (see *Metadata*).

Each building image pair is therefore uniquely identified by:

- the **building ID**
- the **view direction**
- the **earthquake scenario**
- the shape of the **bay grid**

2.2 Metadata

ID	length	width	height	thickness	PGA	POV	Hz
0001	3	4	3	10	0.2458	B	5.091223
0002	4	5	3	10	0.2458	D	4.298888
0003	5	6	3	10	0.2458	A	5.398558
0004	7	8	3	10	0.2458	C	4.298326
0005	8	9	3	10	0.2458	B	3.939938

In addition to the images, each building-earthquake pair is associated with a set of **metadata** describing the structural and seismic parameters. These include:

- **length**, **width**, and **height** of the building (measured in number of bays)
- **wall thickness** (structural thickness of each bay)
- **PGA** (*peak ground acceleration*) — a measure of earthquake intensity
- **POV** (*point of view*) — the view direction (from one of the four sides: A, B, C, or D)
- **Hz** — the dominant frequency of the ground motion

2.3 Model Features

The metadata variables are used as **predictors** (**X**) in the machine learning model, providing critical context about both the building's geometry and the seismic input. They enable the CNN to learn how different structural configurations and earthquake characteristics affect the resulting stress distribution.

While the metadata serve as the input features for the predictive model, the **post-earthquake images**—which show the stress distribution—constitute the **target variables** (**y**). These images provide the ground truth output that the CNN is trained to predict.

In contrast, the **pre-earthquake images** are **not** used as inputs to the model, but just included for preprocessing purposes (see *Data Preprocessing*).

2.4 Data Preprocessing

Predicting the full stress map of a building as a single image is a highly complex task, due to the high dimensionality of the output and the variability in structural layouts. To reduce this complexity and make the learning problem tractable, we adopt a **per-bay** prediction strategy. That is, rather than predicting the entire post-earthquake image at once, the model is trained to predict the stress at the level of individual bays. These local predictions can then be reassembled to reconstruct the complete stress map.

Additionally, note that the images—like the examples shown above—often contain various artifacts that our model is **not** expected to predict. These include:

- **ticks** from the finite element mesh
- **labels** or annotations from the visualization tool
- **slightly irregular** or non-straight grid lines
- **artifacts** introduced by the earthquake simulator (e.g., white boxes)
- **inconsistent** bay sizes in pixel dimensions

These elements are removed or reduced during preprocessing: the pipeline is specifically designed to filter out such noise and standardize the bay regions. This ensures that the model focuses solely on learning the meaningful stress patterns, not irrelevant visual distortions.

–

The preprocessing happens via a Computer Vision (CV) pipeline which automatically identifies the bays on the simpler pre-earthquake images, then applies the same segmentation to the post-earthquake images.

The process begins with each raw input image, where we first isolate the structural content by filtering out background pixels and cropping to the bounding region of the building. This ensures that the analysis focuses exclusively on the meaningful geometry.

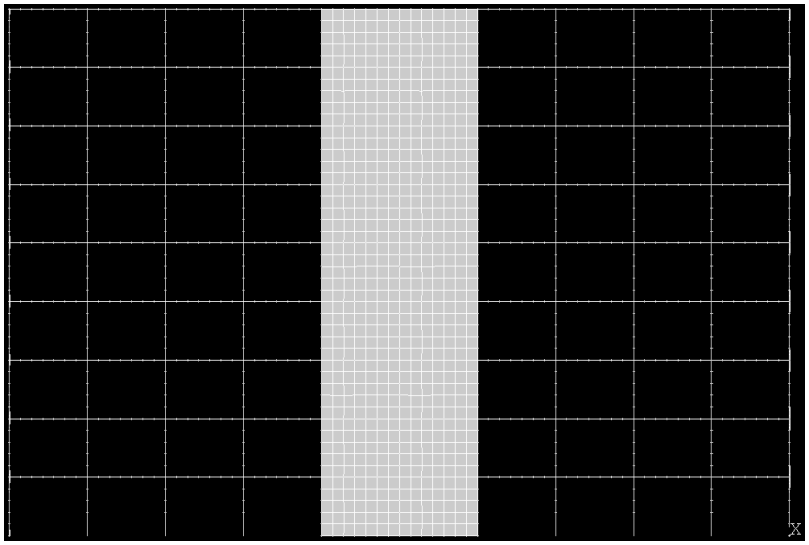

Fig. 1: Step 1 — Filter out non-structural pixels and crop to the relevant building region.

Next, we detect the underlying bay grid by identifying the most prominent vertical and horizontal edges.

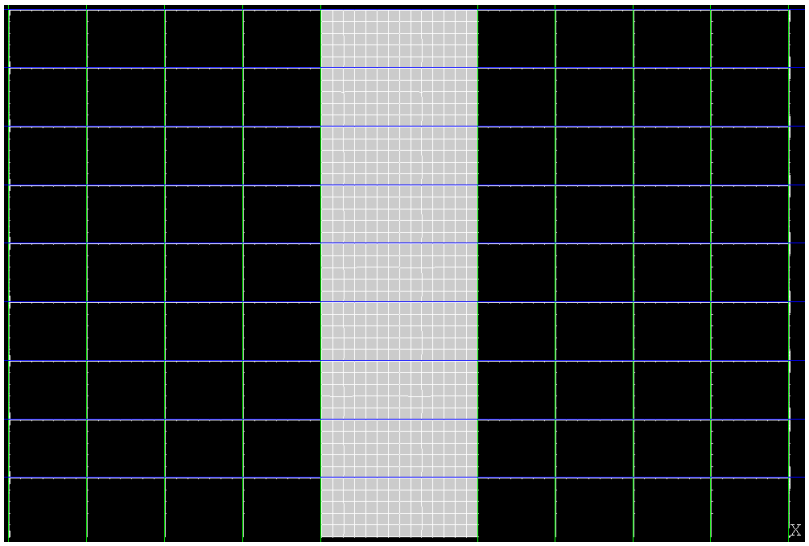

Fig. 2: Step 2 — Detect bay grid layout using edge detection.

The next steps involve extracting a template from the top-left cell of the grid, which serves as a reference for the bay

structure. First, we identify the upper-left intersection points of the detected grid edges.

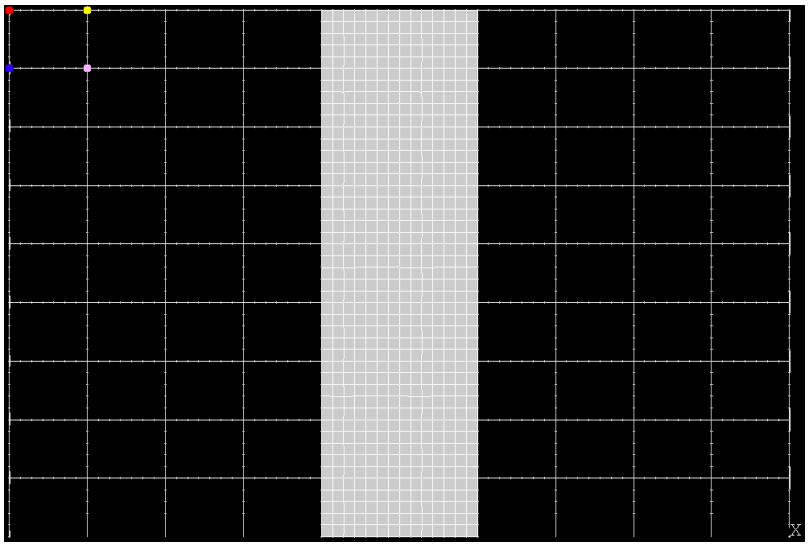

Fig. 3: Step 3 — Identify the upper-left intersection points between the detected grid edges.

Next, a template is extracted from the top-left bay, using the previously identified intersection points.

Fig. 4: Step 4 — Extract a template bay region from the top-left corner of the grid.

Template matching is then used to locate all other bay regions that resemble the extracted template.

From the matched grid, we compute a bounding box that encloses the full bay layout.

We then draw a structured grid of rectangles, each with dimentions equal to those of the template. This ensures a consistent segmentation into cells of equal size. Hower, the cell grid is only an approximation of the actual bay layout, since the bays may not be perfectly aligned or may vary slightly in size.

To capture the actual content of the bays, and avoid the grid lines and ticks, we first slightly shrunk the grid cells.

–

This grid defines the effective segmentation of the image into individual proxy-bay regions, which we can then apply to the corresponding post-earthquake images. While this process does result in minor loss of edge information around each bay, it allows us to focus on the core structural content and avoid artifacts such as grid lines and mesh ticks.

After extraction, each bay image is resized to match the original template shape via bicubic interpolation. While this does **not** preserve the exact size of every bay in the image—since some may vary by a few pixels—it provides a consistent target size across all samples. For our purposes, this caveat is acceptable, as it ensures uniformity in the training dataset.

For the post-earthquake image example shown at the top, the cropped bays of interest extracted from the bottom row appear as shown below:

To reduce high-frequency noise such as the grey pixels and the artifact segments visible in the images above, a Gaussian blur is applied.

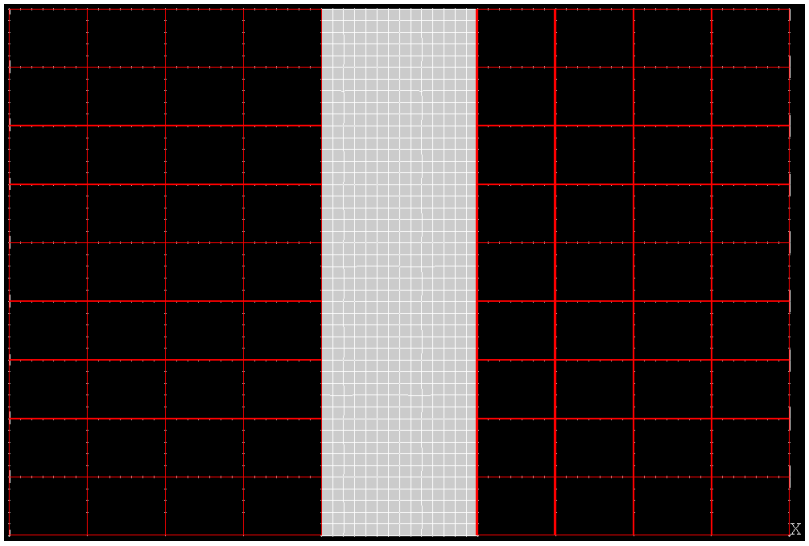

Fig. 5: Step 5 — Detect all bay regions by matching the template across the image.

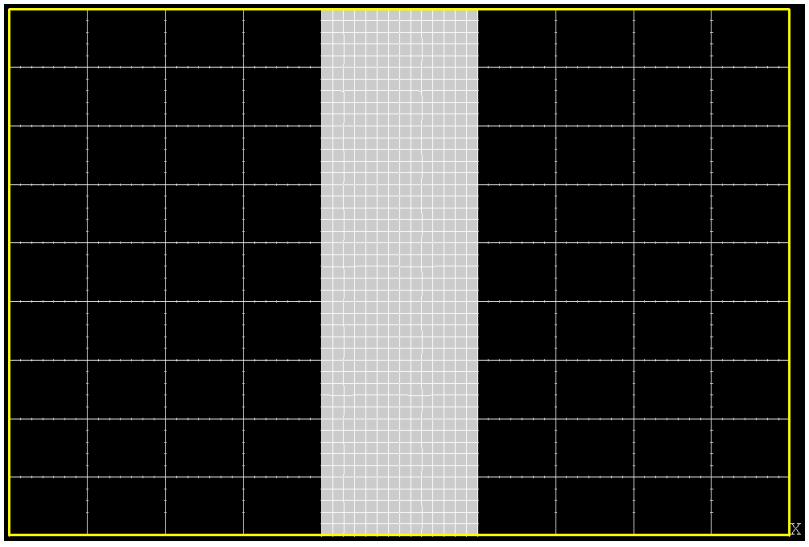

Fig. 6: Step 6 — Draw a bounding box.

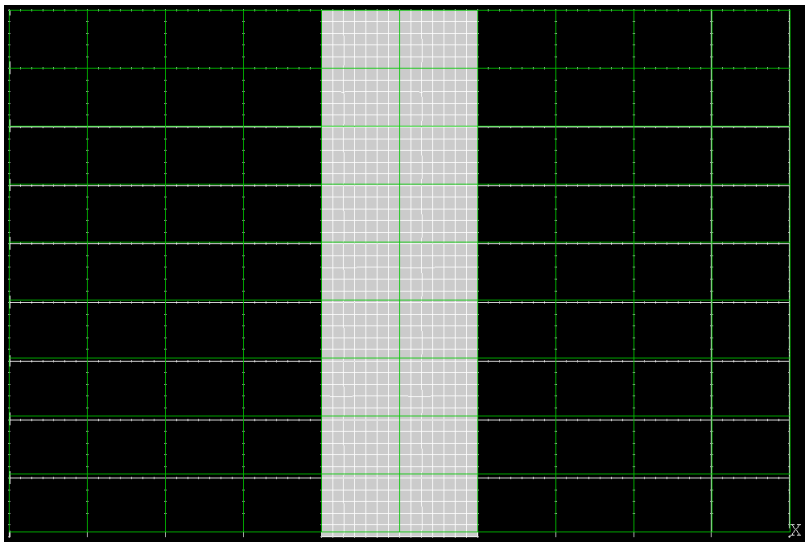

Fig. 7: Step 7 — Draw a uniform grid.

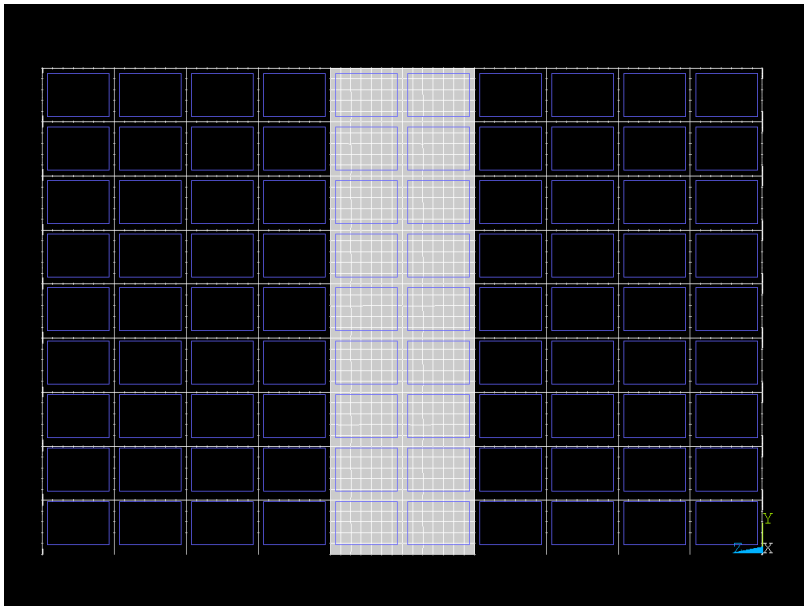

Fig. 8: Step 8 — Shrink all the cells.

This is ultimately followed by a median filtering aimed at restoring structural detail.

Finally, the dark bays—not used in the analysis (such as the first and last 4 columns in the pre-earthquake image example above)—are excluded from the dataset.

–

The result of this pipeline is a clean, well-aligned dataset of labeled bay-level image samples, which can be used to train a deep learning model. This strategy allows us to frame the problem as a structured, supervised learning task without the complexity of generating entire stress maps in one shot.

The predicting variabels are assembled from the metadata introduced in *Metadata* with two minor preprocessing steps.

First, the **POV** feature is converted from a categorical variable to a numerical format using **one-hot encoding** (**OHE**), resulting in three new binary columns: POV_A, POV_B, and POV_C. The POV_D category is intentionally omitted to serve as the reference class and to avoid introducing artificial correlations in the data.

Secondly, since we are considering the images per-bay, we add two extra columns, **r** and **c**, indicating the row and column indices of each bay within the building grid, allowing the model to learn spatial relationships between adjacent bays.

The resulting metadata table, which serves as the input features for the machine learning model, appears as follows:

image	length	width	height	thick-ness	r	c	PGA	POV_A	POV_B	POV_C	Hz
build-ing_1_A_bay_1	3	4	3	10	1	2	0.2458	1	0	0	5.091223
build-ing_1_A_bay_2	3	4	3	10	0	2	0.2458	1	0	0	5.091223
build-ing_1_D_bay_1	3	4	3	10	2	2	0.2458	0	0	0	5.091223
build-ing_5_A_bay_1	4	5	3	10	1	2	0.2458	1	0	0	4.298888
build-ing_5_B_bay_1	4	5	3	10	0	2	0.2458	0	1	0	4.298888

2.5 Data Summary

After preprocessing and metadata integration, the dataset is organized into two components:

- **X**: the input feature matrix, including geometric and seismic metadata along with one-hot encoded orientation.
- **y**: the target data, consisting of images representing stress distributions for individual bays.

Starting from 3,421 buildings, each captured from four different viewpoints, we process a total of approximately 14,000 images. From these, the pipeline extracts roughly **35,000 individual bay regions**, which serve as the model samples.

The following table summarizes the shapes of the input **X** and targets **y**:

Name	Shape	Data Type
X	(~35,000, 11)	Metadata vector
y	(~35,000, height, width, 3)	RGB image (target)

CHAPTER

THREE

CONVOLUTIONAL NEURAL NETWORK MODEL

3.1 Data loading and batching

The training data consists of stress image patches for individual bays, paired with a conditioning vector as described in *Data and Preprocessing*. The conditioning vectors, representing the input **X matrix**, include:

- the building geometry (length, width, height, wall thickness)
- seismic parameters (PGA, dominant frequency)
- viewpoint (encoded via one-hot vectors)
- bay location indices (row and column within the bay grid)

The data is handled using a custom PyTorch *Dataset* class that:

- Loads the conditioning vectors
- Normalizes the numeric variables using a *MinMaxScaler*
- Preserves the one-hot encoded POV variables
- Loads the corresponding post-earthquake RGB image from disk
- Resizes each image to 64 × 64 and normalizes it to the [0, 1] range

The dataset is split into three subsets using an 80/10/10 split.

3.2 Model architecture (PyTorch)

Our predictive model is a **convolutional decoder** that generates the post-earthquake stress image of a single building bay, conditioned on a vector of input parameters.

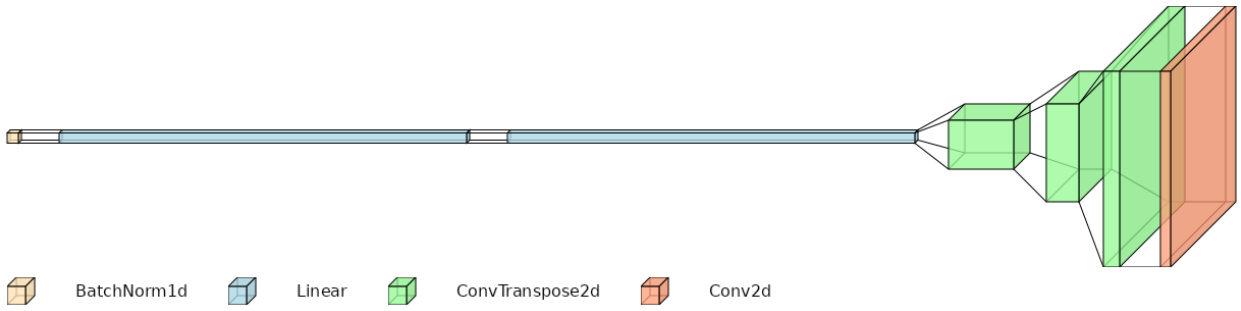

Fig. 1: Model architecture. Diagram created with VisualTorch.

The model effectively performs a **regression** from the conditioning vector to a full-resolution RGB image, using two main stages:

1. A **fully connected projection block** transforms the input conditioning vector into a 2D feature map

2. A stack of **transposed convolutional layers** (a.k.a. "upsampling layers") upsamples this feature map to the desired image resolution

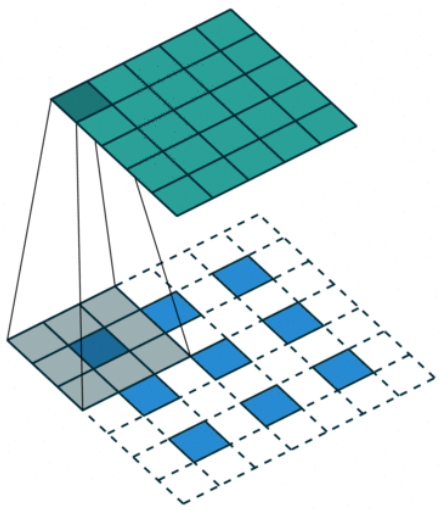

Fig. 2: Figure adapted from conv_arithmetic repository

The conditioning vector is first normalized via **BatchNorm1d**, then passed through two fully connected layers with ReLU activations. The output of these layers is reshaped into a low-resolution 2D feature map with a high number of channels.

This feature map is progressively upsampled by a sequence of **ConvTranspose2d** layers, which increase the spatial resolution while reducing the number of channels. Finally, a **Conv2d** layer with Sigmoid activation maps the upsampled features to the desired 3-channel RGB output, normalized to the [0, 1] range.

3.3 Training

The model is trained for 1000 epochs to minimize the difference between the predicted stress images and the ground truth post-earthquake images using a regression loss function, namely the Mean Squared Error (MSE). Early stopping based on validation loss is applied to retain the best model checkpoint.

Training uses the Adam optimizer with a relatively low learning rate of 0.005 and a large batch size of 256.

Training is performed on all bays extracted from all points of view. Because the viewpoint (POV) is included in the conditioning vector, the model can directly learn to reproduce diverse stress patterns specific to each perspective.

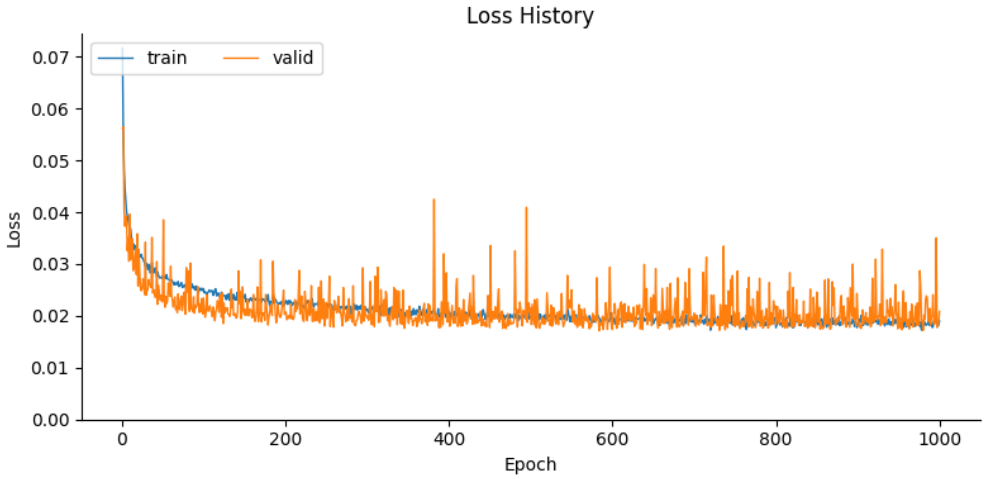

Fig. 3: Training and validation loss over epochs, illustrating the model's convergence.

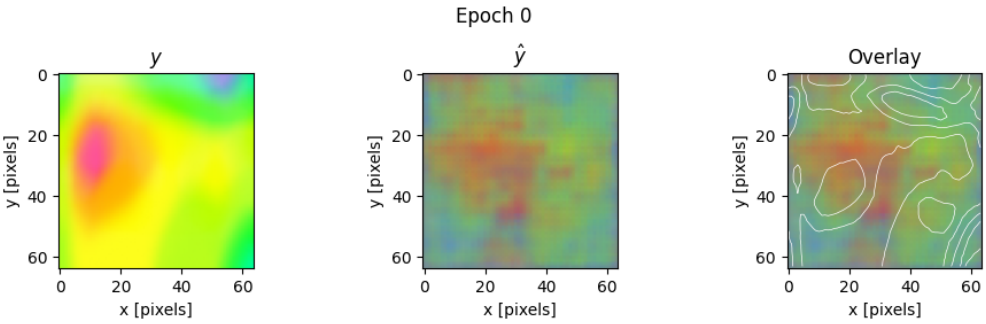

Fig. 4: Example prediction from the validation set at epoch 0, showing the ground truth (left), model output (center), and overlay comparison (right).

CHAPTER

FOUR

MODEL PREDICTIONS

The model achieves a Mean Squared Error (MSE) of approximately 0.017 on the test set, corresponding to a Root Mean Squared Error (RMSE) of about 13.0% relative to the normalized image intensity range [0, 1].

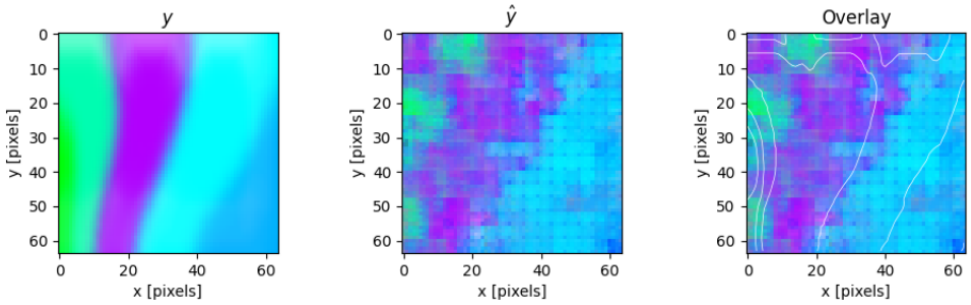

Fig. 1: Example of prediction for a single bay, before assembling, showing the ground truth (left), the model prediction (center), and an overlay comparison (right).

Notice that the predictions are slightly pixellated, which is a common artifact of the upsampling process in convolutional neural networks. This is due to the model generating a low-resolution feature map that is then upsampled to the full resolution, which can introduce some quantization effects.

This effect is partially mitigated during the reconstruction of the building from individual bays (*Building Reconstruction*).

4.1 Building Reconstruction

To visualize the predicted stress for an entire building, the model outputs for each individual bay are first reshaped to match the size and layout of the original bay template. This reshaping happens via bicubic interpolation, which partially smooths out the pixellated artifacts introduced during the upsampling process.

These predicted bay images are then systematically reassembled according to their spatial positions within the building's grid, effectively reconstructing the full post-earthquake stress map of the building.

Similarly, the ground truth bay images are reshaped and arranged in the same manner to reconstruct the ground truth building. This is necessary because the original bays have slightly different sizes and aspect ratios, which can lead to minor variations in the pixel dimensions of the reconstructed images. Reshaping all of them to a common template size allows for a direct visual comparison between predictions and ground truth at the building level.

–

The following images show the post-earthquake stress distribution reconstructed at the building level, alongside the model prediction. The example refers to the same building shown in *Data and Preprocessing*.