Automated Assessment of Residual Plots with Computer Vision Models

Weihao Li[a,b], Dianne Cook[a], Emi Tanaka[b], Susan VanderPlas[c], Klaus Ackermann[a]

[a]Department of Econometrics and Business Statistics, Monash University, Clayton, VIC, Australia; [b]Research School of Finance, Actuarial Studies and Statistics, Australian National University, Acton, ACT, Australia; [c]Department of Statistics, University of Nebraska, Lincoln, Nebraska, USA

ARTICLE HISTORY

Compiled February 19, 2026

ABSTRACT

Plotting the residuals is a recommended procedure to diagnose deviations from linear model assumptions, such as non-linearity, heteroskedasticity, and non-normality. The presence of structure in residual plots can be tested using the lineup protocol to do visual inference. There are a variety of conventional residual tests, but the lineup protocol, used as a statistical test, performs better for diagnostic purposes because it is less sensitive and applies more broadly to different types of departures. However, the lineup protocol relies on human judgment which limits its scalability. This work presents a solution by providing a computer vision model to automate the assessment of residual plots. It is trained to predict a distance measure that quantifies the disparity between the residual distribution of a fitted classical normal linear regression model and the reference distribution, based on Kullback-Leibler divergence. Evaluation using data from a prior human-subject experiment shows the model exhibits lower sensitivity than conventional tests but higher sensitivity than human visual tests. Several examples from classical papers and contemporary data demonstrate the method's practical utility in automating residual diagnostics.

KEYWORDS

statistical graphics; data visualization; visual inference; computer vision; machine learning; hypothesis testing; regression analysis; cognitive perception; simulation; practical significance

1. Introduction

Plotting residuals is commonly regarded as a standard practice in linear regression diagnostics (Belsley, Kuh, and Welsch 1980; Cook and Weisberg 1982). This visual assessment plays a crucial role in identifying whether key model assumptions, such as linearity, homoscedasticity, and normality, are reasonable. It also helps in understanding the goodness of fit and various unexpected characteristics of the model.

Generating a residual plot in most statistical software is often as straightforward as executing a line of code or clicking a button. However, accurately interpreting a residual plot can be challenging, as some features are a result of predictor characteristics and natural variation, while others indicate violations of model assumptions (Li et al. 2024). For example, in Figure 1, the residual plot displays a triangular left-pointing shape. The distinct difference in the spread of the residuals across the fitted values might lead analysts to suspect heteroskedasticity. In this case, however, the fitted regression model is correctly specified, and the triangular shape is actually a result of the skewed distribution of the predictors, rather than indicating a flaw in the model.

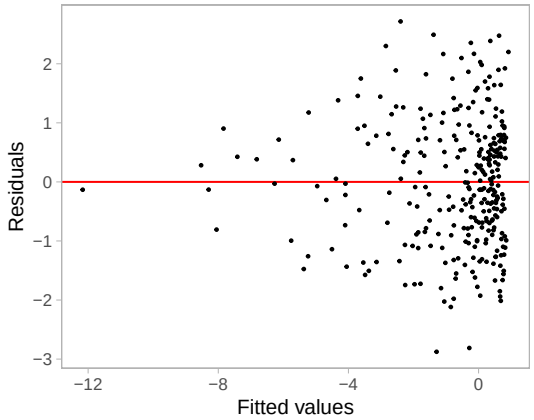

Figure 1. An example residual vs fitted values plot (red line indicates 0 corresponds to the x-intercept, i.e. $y = 0$). The vertical spread of the data points varies with the fitted values. This often indicates the existence of heteroskedasticity, however, here the result is due to skewed distribution of the predictors rather than heteroskedasticity. The Breusch-Pagan test rejects this residual plot at 95% significance level (p-value = 0.046).

The concept of visual inference, as proposed by Buja et al. (2009), provides an inferential framework to assess whether residual plots indeed contain visual patterns inconsistent with the model assumptions. The fundamental idea involves testing whether the true residual plot visually differs significantly from the null plots, where null plots

are plotted with residuals generated from the residual rotation distribution (Langsrud 2005), which is a distribution consistent with the null hypothesis H_0 that the linear regression model is correctly specified. Typically, the visual test is accomplished through the lineup protocol (Figure 2), where the true residual plot is embedded within a field of null plots. If the true residual plot can be distinguished from the lineup, it provides evidence for rejecting H_0.

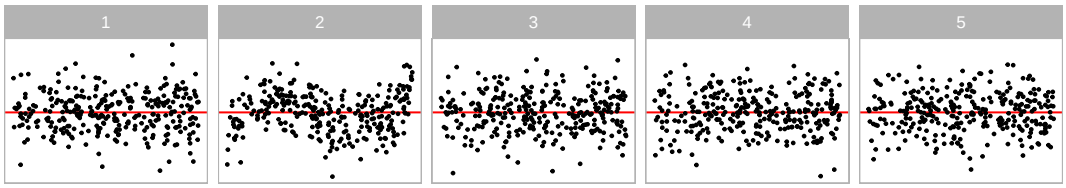

Figure 2. An example lineup embedding the true residual plot among four null plots. In practice, lineups typically include 19 null plots, but a reduced set is shown here for presentation purposes. The null plots are generated via residual rotation to ensure consistency with H_0. Observers who have not previously seen the lineup are asked to identify the plot that appears most different. Under H_0 that the regression model is correctly specified, the true residual plot should be indistinguishable from the null plots, yielding a selection probability of 0.2. A small p-value arises when a substantial proportion of observers select the true residual plot (shown at position 2, exhibiting non-linearity).

The practice of delivering a residual plot as a lineup is generally regarded as a valuable approach. Beyond its application in residual diagnostics, the lineup protocol has been integrated into the analysis of diverse subjects. For instance, Loy and Hofmann (2013; 2014; 2015) illustrated its applicability in diagnosing hierarchical linear models. Additionally, Widen et al. (2016) and Fieberg, Freeman, and Signer (2024) demonstrated its utility in geographical and ecology research respectively, while Krishnan and Hofmann (2021) explored its effectiveness in forensic examinations.

A practical limitation of the lineup protocol lies in its reliance on human judgments (see Li et al. 2024). Unlike conventional statistical tests that can be performed using in statistical software, the lineup protocol requires human evaluation of images. This characteristic makes it less suitable for large-scale applications, given the associated high labor costs and time requirements. There is a substantial need to develop an approach to substitute these human judgement with an automated reading of data plots using computer vision.

Efforts to automate plot interpretation date back to early work like Tukey's "Scagnostics" (Tukey and Tukey 1985), which is a set of numerical statistics that summarize features of scatter plots. Wilkinson, Anand, and Grossman (2005) expanded

on this work, introducing scagnostics based on computable measures applied to planar proximity graphs. These measures, including, but not limited to, "Outlying", "Skinny", "Stringy", "Straight", "Monotonic", "Skewed", "Clumpy", and "Striated", aimed to characterize outliers, shape, density, trend, coherence and other characteristics of the data. However, such measures may not capture the full range of visual patterns necessary for residual diagnostics (Buja et al. 2009). A more promising direction lies in enabling computers to learn the relevant visual features directly mirroring how humans intuitively interpret plots.

Modern computer vision models rely on deep neural networks with convolutional layers (Fukushima and Miyake 1982). These layers use small, sliding windows to scan the image, performing a dot product to extract local features and patterns. Numerous studies have demonstrated the efficacy of convolutional layers in addressing various vision tasks, including image recognition (Rawat and Wang 2017). Despite the widespread use of computer vision models in fields like computer-aided diagnosis (Lee and Chen 2015), pedestrian detection (Brunetti et al. 2018), and facial recognition (Emami and Suciu 2012), their application in reading data plots remains limited. While some studies have explored the use of computer vision models for tasks such as reading recurrence plots for time series regression (Ojeda, Solano, and Peramo 2020), time series classification (Chu et al. 2019; Hailesilassie 2019; Hatami, Gavet, and Debayle 2018; Zhang et al. 2020), anomaly detection (Chen, Su, and Yang 2020), and pairwise causality analysis (Singh et al. 2017), the application of reading residual plots with computer vision models is a new field of study.

In this paper, we develop computer vision models and integrate them into the residual plot diagnostics workflow, addressing the need for automated visual inference. The model is a regression-based computer vision system that takes a residual plot as input and outputs a non-negative estimated distance $\hat{D}$. Training and test data are generated from a synthetic data model, simulating residual plots under various violation scenarios, with targets defined as customized distances D computed from the underlying data generating process, quantifying the extent of model violations. The model performance is evaluated on held-out test plots and compared with classical residual diagnostic tests as well as results from a human-subject experiment.

The paper is structured as follows. Section 2 discusses various specifications of the computer vision models. Section 3 defines the distance measure used to detect model violations, while Section 4 explains how the computer vision models estimate this distance measure. Section 5 covers the statistical tests based on the estimated distance. Sections 6 and 7 describe the model architecture and the corresponding data generation and training process, respectively. The results are presented in Section 8. Example dataset applications are discussed in Section 9. Finally, we conclude with a discussion of our findings and propose ideas for future research directions.

2. Model Specifications

There are various specifications of the computer vision model that can be used to assess residual plots. We discuss these specifications below focusing on the input and the output formats.

2.1. *Input Formats*

Deep learning models are sensitive to input design, and several alternatives are available for encoding residual plots. A simple approach involves feeding vectors of residuals and fitted values into the model. While this format contains all relevant information, the input length varies with sample size, which conflicts with the fixed-shape requirements of modern vision models such as those implemented in TensorFlow (Abadi et al. 2016). Padding can be used to enforce uniform length but risks truncation if inputs exceed the fixed size. Alternatively, fixed-size sampling of residual–fitted value pairs can ensure consistency, though at the cost of some information loss.

Another strategy is to convert residual plots into images. This enables the use of standard convolutional architectures in computer vision such as VGG16 (Simonyan and Zisserman 2014). Image-based inputs offer a richer representation by preserving the joint distribution of residuals and fitted values in a visually interpretable format, helping models more effectively capture spatial structure and local correlations than vector-based formats. To ensure model generalization, it is essential to use residual plots in a consistent style, i.e., same aesthetics, layout, scaling and color schemes.

Using multiple residual plots as input, such as pairs, triplets, or full lineups is also a possible option. For instance, triplet networks (Chopra, Hadsell, and LeCun 2005) can assess visual similarity if we assume null plots are more similar to each other than to the true residual plot, which may help distinguish null plots from true residual plots. However, these architectures introduce complexity in training due to weight sharing and specialized loss functions. We experimented with this setting in a pilot study, and found multiple residual plots led to high input resolution, high computational cost and suboptimal model performance. Balancing accuracy, interpretability, and implementation cost, we adopt a single residual plot in image format as the model input.

2.2. *Output Formats*

Given the input is a single fixed-resolution image, the model output can be defined for various tasks, including binary classification, multiclass classification, and numeric regression.

In the binary case, the outcome may indicate whether the input image is consistent with a null plot, as determined either by (1) the data-generating process or (2) the outcome of a visual test based on human judgment. The first approach models the probability that a residual plot is not a null plot and is a viable option. In contrast, the second relies on experimental data that are often scarce or inconsistent across studies.

Alternatively, a multiclass outcome may classify images into categories such as "contains outliers" vs. "does not contain outliers" or "is non-linear" vs. "not non-linear," among other diagnostic features. While this approach is appealing in theory, it essentially reverts to the traditional testing framework, where tests focus on detecting specific types of model violations in isolation.

A third option is to produce a meaningful and interpretable numerical measure that quantifies the strength of suspicious visual patterns reflecting the extent of model violations, or the difficulty index for identifying whether a residual plot has no issues. However, such measures are often used informally in daily communications but are rarely formalized. For supervised learning, they must be clearly defined and reliably

estimable.

In this study, we define and use a continuous distance between a true residual plot and a theoretically "good" residual plot. This approach captures the degree to which a plot deviates from model assumptions. In contrast, a probability output based on binary labels, where true residual plots are labeled as 1 and null plots as 0, cannot express the severity of violation. It forces the model to infer this information solely from the data distribution, which may be inefficient and discards useful prior knowledge.

3. Distance from a Theoretically "Good" Residual Plot

To develop a computer vision model for assessing residual plots within the visual inference framework, it is important to precisely define a numerical measure of "difference" or "distance" between plots. This distance can take the form of a basic statistical operation on pixels, such as the sum of square differences, however, a pixel-to-pixel comparison makes little sense in comparing residual plots where the main interest would be structural patterns. Alternatively, it can involve established image similarity metrics like the Structural Similarity Index Measure (Wang et al. 2004), which compares images by integrating three perception features of an image: contrast, luminance, and structure (related to average, standard deviation and correlation of pixel values over a window, respectively). These image similarity metrics are tailored for image comparison in tasks that are vastly different from evaluating data plots, where only essential plot elements require assessment (Chowdhury et al. 2018). A different direction is to define a notion of distance by integrating key plot elements (instead of key perception features like luminance, contrast, and structure), such as those captured by scagnostics (see Section 1), but the functional form needs to be carefully tailored to reflect model violations meaningfully.

In this section, we introduce a distance measure between a true residual plot and a theoretically "good" residual plot. This measure quantifies the divergence between the residual distribution of a given fitted regression model and that of a correctly specified model. The computation assumes knowledge of the data generating processes for predictors and response variables. Since these processes are often unknown in

practice, we will discuss a method to estimate this distance using a computer vision model in Section 4.

3.1. *Residual Distribution*

For a classical normal linear regression model, $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e}$, the residuals $\hat{\boldsymbol{e}}$ are derived as the difference of the fitted values and observed values $\boldsymbol{y}$. Suppose the data generating process is known and the regression model is correctly specified, by the Frisch-Waugh-Lowell theorem (Frisch and Waugh 1933), residuals $\hat{\boldsymbol{e}}$ can also be treated as random variables and written as a linear transformation of the error $\boldsymbol{e}$ formulated as $\hat{\boldsymbol{e}} = \boldsymbol{R}\boldsymbol{e}$, where $\boldsymbol{R} = \boldsymbol{I}_n - \boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top$ is the residual operator, $\boldsymbol{I}_n$ is a n by n identity matrix, and n is the number of observations.

One of the assumptions of the classical normal linear regression model is that the error $\boldsymbol{e}$ follows a multivariate normal distribution with zero mean and constant variance, i.e., $\boldsymbol{e} \sim N(\boldsymbol{0}_n, \sigma^2 \boldsymbol{I}_n)$. It follows that the distribution of residuals $\hat{\boldsymbol{e}}$ can be characterized by a certain probability distribution, denoted as Q, which is transformed from the multivariate normal distribution. This reference distribution Q summarizes what "good" residuals should follow given the design matrix $\boldsymbol{X}$ is known and fixed.

Suppose the design matrix $\boldsymbol{X}$ has linearly independent columns, the trace of the hat matrix $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top$ equals the number of columns in $\boldsymbol{X}$ denoted as k. As a result, the rank of $\boldsymbol{R}$ is $n-k$, and Q is a degenerate multivariate distribution. To capture the characteristics of Q, such as moments, we can simulate a large number of $\boldsymbol{\varepsilon}$ and transform it to $\boldsymbol{e}$ to get the empirical estimates. For the purpose of this study, we replaced the variance-covariance matrix of residuals $\mathrm{cov}(\boldsymbol{e}, \boldsymbol{e}) = \boldsymbol{R}\sigma^2\boldsymbol{R}^\top = \boldsymbol{R}\sigma^2$ with a full-rank diagonal matrix $\mathrm{diag}(\boldsymbol{R}\sigma^2)$, where diag(.) sets the non-diagonal entries of a matrix to zeros. The resulting distribution for Q was $N(\boldsymbol{0}_n, \mathrm{diag}(\boldsymbol{R}\sigma^2))$. This replacement ensures a non-degenerate distribution, which is required for computing the distance defined in Section 3.2.

Distribution Q is derived from the correctly specified model. However, if the model is misspecified, then the actual distribution of residuals denoted as P, will be different from Q. For example, if the data generating process contains variables correlated with any column of $\boldsymbol{X}$ but are missing from $\boldsymbol{X}$, causing an omitted variable problem, P will

be different from Q because the residual operator obtained from the fitted regression model will not be the same as $\boldsymbol{R}$. Besides, if the $\boldsymbol{\varepsilon}$ follows a non-normal distribution such as a multivariate lognormal distribution, P will usually be skewed and has a long tail.

3.2. *Distance of P from Q*

Defining a proper distance between distributions is usually easier than defining a proper distance between data plots. Given the true residual distribution P and the reference residual distribution Q, we used a distance measure based on Kullback-Leibler divergence (Kullback and Leibler 1951) to quantify the difference between two distributions as

$$D = \log\left(1 + D_{KL}(P\|Q)\right), \tag{1}$$

where $D_{KL}(P\|Q)$ is defined as

$$D_{KL}(P\|Q) = \int_{\mathbb{R}^n} \log \frac{p(\boldsymbol{e})}{q(\boldsymbol{e})} p(\boldsymbol{e}) d\boldsymbol{e}, \tag{2}$$

and $p(.)$ and $q(.)$ are the probability density functions for distribution P and distribution Q, respectively.

This distance measure was first proposed in Li et al. (2024), which was mainly designed for measuring the effect size of non-linearity and heteroskedasticity in a residual plot. Li et al. (2024) has derived that, for a classical normal linear regression model that omits necessary higher-order predictors $\boldsymbol{Z}$ and the corresponding parameter $\boldsymbol{\beta}_z$, while incorrectly assuming $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}_n, \sigma^2 \boldsymbol{I}_n)$, but in fact $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}_n, \boldsymbol{V})$ where $\boldsymbol{V}$ is an arbitrary symmetric positive semi-definite matrix, P can be represented as $N(\boldsymbol{RZ\beta}_z, \text{diag}(\boldsymbol{RVR}))$. Note that the variance-covariance matrix was replaced with the diagonal matrix to ensure it is a full-rank matrix.

Since both P and Q are adjusted to be multivariate normal distributions, Equation

2 can be further expanded to

$$D_{KL}(P\|Q) = \frac{1}{2}\left(\log\frac{|\boldsymbol{W}|}{|\text{diag}(\boldsymbol{R}\sigma^2)|} - n + \text{tr}(\boldsymbol{W}^{-1}\text{diag}(\boldsymbol{R}\sigma^2)) + \boldsymbol{\mu}_z^\top \boldsymbol{W}^{-1}\boldsymbol{\mu}_z\right), \quad (3)$$

where $\boldsymbol{\mu}_z = \boldsymbol{RZ\beta}_z$, and $\boldsymbol{W} = \text{diag}(\boldsymbol{RVR})$. The assumed error variance σ^2 is set to be $\text{tr}(\boldsymbol{V})/n$, which is the expectation of the estimated variance.

3.3. *Non-normal P*

For non-normal error $\boldsymbol{\varepsilon}$, the true residual distribution P is unlikely to be a multivariate normal distribution. Thus, Equation 3 given in Li et al. (2024) will not be applicable to models violating the normality assumption.

To evaluate the Kullback-Leibler divergence of non-normal P from Q, the fallback is to evaluate Equation 2 numerically. However, since $\boldsymbol{e}$ is a linear transformation of non-normal random variables, it is very common that the general form of P is unknown, meaning that we can not easily compute $p(\boldsymbol{e})$ using a well-known probability density function. Additionally, even if $p(\boldsymbol{e})$ can be calculated for any $\boldsymbol{e} \in \mathbb{R}^n$, it will be very difficult to do numerical integration over the n-dimensional space, because n could be potentially very large.

In order to approximate $D_{KL}(P\|Q)$ in a practically computable manner, the elements of $\boldsymbol{e}$ are assumed to be independent of each other. This assumption solves both of the issues mentioned above. First, we no longer need to integrate over n random variables. The result of Equation 2 is now the sum of the Kullback-Leibler divergence evaluated for each individual residual due to the assumption of independence between observations. Second, it is not required to know the joint probability density $p(\boldsymbol{e})$ any more. Instead, the evaluation of Kullback-Leibler divergence for an individual residual relies on the knowledge of the marginal density $p_i(e_i)$, where e_i is the i-th residual for $i = 1, ..., n$. This is much easier to approximate through simulation. It is also worth mentioning that this independence assumption generally will not hold if $\text{cov}(e_i, e_j) \neq 0$ for any $1 \leq i < j \leq n$, but its existence is essential for reducing the computational cost.

Given $\boldsymbol{X}$ and $\boldsymbol{\beta}$, the algorithm for approximating Equation 2 starts from simulating m sets of observed values $\boldsymbol{y}$ according to the data generating process. The observed values are stored in a matrix $\boldsymbol{A}$ with n rows and m columns, where each column of $\boldsymbol{A}$ is a set of observed values. Then, we can get m sets of realized values of $\boldsymbol{e}$ stored in the matrix $\boldsymbol{B}$ by applying the residual operator $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{A}$. Furthermore, kernel density estimation (KDE) with Gaussian kernel and optimal bandwidth selected by the Silverman's rule of thumb (Silverman 2018) is applied on each row of $\boldsymbol{B}$ to estimate $p_i(e_i)$ for $i = 1, ..., n$. The KDE computation can be done by the `density` function in R.

Since the Kullback-Leibler divergence can be viewed as the expectation of the log-likelihood ratio between distribution P and distribution Q evaluated on distribution P, we can reuse the simulated residuals in matrix $\boldsymbol{B}$ to estimate the expectation by the sample mean. With the independence assumption, for non-normal P, $D_{KL}(P\|Q)$ can be approximated by

$$D_{KL}(P\|Q) \approx \sum_{i=1}^{n} \hat{D}_{KL}^{(i)}(P\|Q), \quad \hat{D}_{KL}^{(i)}(P\|Q) = \frac{1}{m}\sum_{j=1}^{m} \log \frac{\hat{p}_i(B_{ij})}{q(B_{ij})},$$

where $\hat{D}_{KL}^{(i)}(P\|Q)$ is the estimator of the Kullback-Leibler divergence for an individual residual e_i, B_{ij} is the i-th row and j-th column entry of the matrix $\boldsymbol{B}$, $\hat{p}_i(.)$ is the kernel density estimator of $p_i(.)$, $q(.)$ is the normal density function with mean zero and an assumed variance estimated as $\hat{\sigma}^2 = \sum_{b\in vec(\boldsymbol{B})}(b - \sum_{b\in vec(\boldsymbol{B})} b/nm)^2/(nm-1)$, and $vec(.)$ is the vectorization operator which turns a $n \times m$ matrix into a $nm \times 1$ column vector by stacking the columns of the matrix on top of each other.

4. Distance Estimation

We previously defined a distance measure (Equation 1) to quantify the difference between the true residual distribution P and an ideal reference distribution Q. However, this distance measure can only be computed when the data generating process is known. In reality, we often have no knowledge about the data generating process,

otherwise we do not need to do a residual diagnostic in the first place.

To estimate this distance from a residual plot, we proposed a computer vision estimator $\hat{D}$ formulated as

$$\hat{D} = f_{CV}(V_{h\times w}(\boldsymbol{e}, \hat{\boldsymbol{y}})),$$

where $V_{h\times w}(.)$ renders the residuals vs. fitted values plot as an $h \times w$ RGB image, and $f_{CV}(.)$ maps the image to an estimated distance $\hat{D} \in [0, +\infty)$. Details of the model architecture are provided in Section 6, and the training procedure is described in Section 7.

The distance estimator $\hat{D}$ allows us to assess how closely the residuals resemble an ideal distribution and use as an index of model violation severity (see Appendix D for details). However, $\hat{D}$ is not expected to equal the true distance D, as a single residual plot may not capture all characteristics of the residual distribution. For the same distribution P, simulated plots can vary visually, especially with small n, leading to estimation error that depends on how representative the input plot is.

5. Statistical Testing

5.1. *Lineup Evaluation*

Theoretically, the distance D for a correctly specified model is 0, as $P = Q$. However, a computer vision model may not predict $\hat{D} = 0$ for a null plot. For example, Figure 1 shows a null plot with a pattern suggestive of heteroskedasticity. The model can not discern whether such patterns arise from heteroskedasticity or from skewed fitted values. Additionally, some null plots could have outliers or strong visual patterns due to randomness, and a reasonable model will try to summarize those information into the prediction, resulting in $\hat{D} > 0$. This is not problematic when $\hat{D}$ is large, as strong visual signals typically justify rejection of H_0. But when $\hat{D}$ is moderate, it is insufficient alone for decision-making.

To address this issue we can adhere to the paradigm of visual inference, by compar-

ing the estimated distance $\hat{D}$ to the estimated distances for the null plots in a lineup. Specifically, if a lineup comprises 20 plots, H_0 will be rejected if $\hat{D}$ exceeds the maximum estimated distance among the 19 null plots, denoted as $\max_{1 \leq i \leq 19} \hat{D}^{(i)}_{null}$, where $\hat{D}^{(i)}_{null}$ represents the estimated distance for the i-th null plot. This approach is conceptually equivalent to the typical lineup protocol requiring a 95% significance level, where H_0 is rejected if the data plot is identified as the most distinct plot by the sole observer.

Moreover, if the number of plots in a lineup, denoted by m, is sufficiently large, the empirical distribution of $\hat{D}^{(i)}_{null}$ can be viewed as an approximation to the null distribution of the estimated distance. Consequently, quantiles of the null distribution can be estimated using the sample quantiles and used for decision-making purposes. This follows the standard simulation-based approach for approximating the sampling distribution of a test statistic in traditional statistics. Details of the computation of sample quantiles can be found in Hyndman and Fan (1996). For instance, if $\hat{D}$ is greater than or equal to the 95% sample quantile, denoted as $Q_{null}(0.95)$, we can conclude that the estimated distance for the true residual plot is significantly different from the estimated distance for null plots at the 95% significance level. In practice, at least 100 null plots are typically required for stable estimate of the null distribution, though more may be necessary if the distribution exhibits heavy tails. Alternatively, a p-value can be defined as the probability, under H_0, of observing a distance at least as large as $\hat{D}$, and it can be estimated as $\frac{1}{m} + \frac{1}{m} \sum_{i=1}^{m-1} I\left(\hat{D}^{(i)}_{null} \geq \hat{D}\right)$.

To reduce computational cost, a pre-computed lattice of $\hat{D}$ quantiles under H_0 for various sample sizes can be used. By matching observed values to the closest entries in this lattice, approximate p-values can be obtained efficiently, with minor loss of precision.

5.2. *Bootstrapping*

Bootstrap methods are commonly used in linear regression to estimate parameter variability without strong distributional assumptions (Davison and Hinkley 1997; Efron and Tibshirani 1994). This involves resampling observations with replacement and refitting the model.

Similarly, we can apply bootstrapping to the estimated distance $\hat{D}$. For each boot-

strap sample $i = 1, \ldots, n_{boot}$, we obtain a refitted model $M_{boot}^{(i)}$, a corresponding residual plot $V_{boot}^{(i)}$, and a predicted distance $\hat{D}_{boot}^{(i)}$. If we are interested in the variation of $\hat{D}$, the distribution of $\hat{D}_{boot}^{(i)}$ can be used to construct confidence intervals.

Alternatively, since each $M_{boot}^{(i)}$ has its own residual distribution, a new approximated null distribution can be construed and the corresponding 95% sample quantile $Q_{boot}^{(i)}(0.95)$ can be computed. Then, if $\hat{D}_{boot}^{(i)} \geq Q_{boot}^{(i)}(0.95)$, H_0 will be rejected for $M_{boot}^{(i)}$. The ratio of rejected $M_{boot}^{(i)}$ among all the refitted models reflects how often the assumed regression model are considered to be misspecified if data were repeatedly drawn from the same process. But this approach is computationally very expensive because it requires $n_{boot} \times (n_{null} + 1)$ times of residual plot assessment. In practice, $Q_{null}(0.95)$ can be used to replace $Q_{boot}^{(i)}(0.95)$ in the computation.

6. Model Architecture

The architecture of the computer vision model f_{CV} is adapted from the well-established VGG16 architecture (Simonyan and Zisserman 2014). While more recent architectures like ResNet (He et al. 2016) and DenseNet(Huang et al. 2017), have achieved even greater performance, VGG16 remains a solid choice for many applications due to its simplicity and effectiveness. Our decision to use VGG16 aligns with our goal of starting with a proven and straightforward model. Figure 3 provides a diagram of the architecture. More details about the neural network layers used in this study are provided in the Appendix B.

The model begins with an input layer of shape $n \times h \times w \times 3$, capable of handling n RGB images. This is followed by a grayscale conversion layer using the luma formula under the Rec. 601 standard (Series 2011), which converts the color image to grayscale. Grayscale suffices for our task since data points are plotted in black. We experiment with three combinations of h and w: 32×32, 64×64, and 128×128, aiming to achieve sufficiently high image resolution for the problem at hand.

The processed image is used as the input for the first convolutional block. The model comprises at most five consecutive convolutional blocks, mirroring the original VGG16 architecture. Within each block, there are two 2D convolutional layers followed

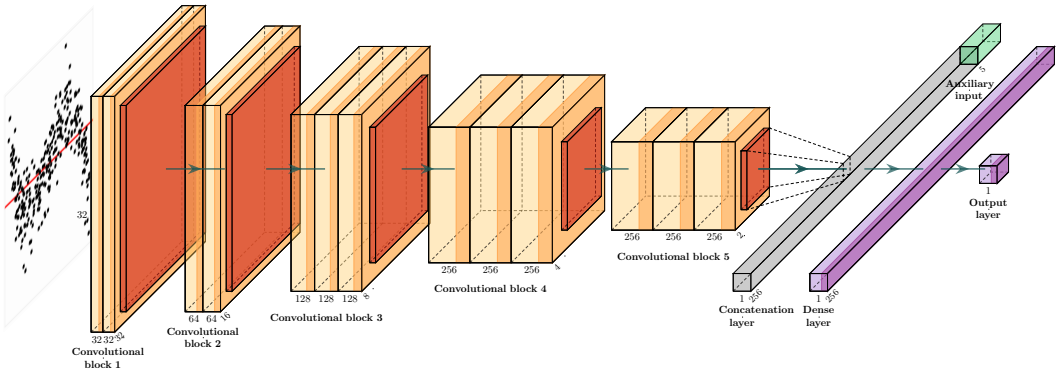

Figure 3. Diagram of the architecture of the optimized computer vision model. Numbers at the bottom of each box show the shape of the output of each layer. The band of each box drawn in a darker color indicates the use of the rectified linear unit activation function. Yellow boxes are 2D convolutional layers, orange boxes are pooling layers, the grey box is the concatenation layer, and the purple boxes are dense layers.

by two activation layers, respectively. Subsequently, a 2D max-pooling layer follows the second activation layer. The 2D convolutional layer convolves the input with a fixed number of 3×3 convolution filters, while the 2D max-pooling layer downsamples the input along its spatial dimensions by taking the maximum value over a 2×2 window for each channel of the input. The activation layer employs the rectified linear unit (ReLU) activation function, a standard practice in deep learning, which introduces a non-linear transformation of the output of the 2D convolutional layer. Additionally, to regularize training, a batch normalization layer is added after each 2D convolutional layer and before the activation layer. Finally, a dropout layer is appended at the end of each convolutional block to randomly set some inputs to zero during training, further aiding in regularization.

The output of the last convolutional block is summarized by either a global max-pooling layer or a global average-pooling layer, resulting in a two-dimensional tensor. To enrich predictions with additional information beyond visual features, this tensor is concatenated with an additional $n \times 5$ tensor, which contains the "Monotonic", "Sparse", "Splines", and "Striped" measures computed using the `cassowaryr` R package (Mason et al. 2022) , along with the number of observations for n residual plots. These measures were selected for their reliability and efficiency, as other scagnostics occasionally caused R process crashes ($\sim 5\%$) during data preparation due to a bug in the `interp` package (Gebhardt, Bivand, and Sinclair 2023). Although this bug was

later fixed at our request, the fix came too late to re-train the model. Moreover, their high computational cost makes them unsuitable for fast inference.

The concatenated tensor is then fed into the final prediction block. This block consists of two fully-connected layers. The first layer contains at least 128 units, followed by a dropout layer. A batch normalization layer is inserted between the fully-connected layer and the dropout layer for regularization purposes. The second fully-connected layer consists of only one unit, serving as the output of the model.

The model weights $\boldsymbol{\theta}$ were randomly initialized using the Glorot Uniform method (Glorot and Bengio 2010) and they were optimized by the Adam optimizer (Kingma and Ba 2014) with the MSE loss function $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (D_i - f_{\boldsymbol{\theta}}(V_i, S_i))^2$, where n_{train} is the number of training samples, V_i is the i-th residual plot and S_i is the auxiliary information about the i-th residual plot including four scagnostics and the number of observations.

7. Data Generation and Model Training

To train the computer vision model for computing $\hat{D}$ (as described in Section 4), we used paired residual plots and their corresponding D values. In deep learning, it is common to train computer vision models with synthetic data, especially when labelled data are scarce, costly, or difficult to annotate reliably (Nikolenko et al. 2021). This approach is particularly suitable when the underlying data-generating process is well understood and large numbers of controlled examples can be produced, allowing the model to learn visual patterns associated with specific violations without the cost and ambiguity of manual labelling. Accordingly, we generated a synthetic dataset consisting of 80,000 training and 8,000 test residual plots, providing full control over the data-generating process, precise computation of D, cost-effective scaling, and coverage of a wide range of visual patterns associated with model violations.

The simulated data incorporated three common types of residual departures: non-linearity, heteroskedasticity, and non-normality. These were introduced by fitting a standard simple linear regression model to data generated from a more flexible model based on Hermite polynomials (Hermite 1864), multiple predictors, and varying dis-

Table 1. Number of training and test samples for each model violation scenario, including cases with multiple simultaneous violations. Each sample consists of a residual plot as input and the corresponding distance D as the target. Root mean square error (RMSE) values for the training and test sets are shown for each scenario. Details of the synthetic data generating process used to construct these samples are provided in Appendix A.

	Train		Test	
Violations	#samples	RMSE	#samples	RMSE
no violations	1,546	1.149	155	1.267
non-linearity	22,184	0.625	2,218	0.787
heteroskedasticity	10,826	0.528	1,067	0.602
non-linearity + heteroskedasticity	10,221	0.593	985	0.751
non-normality	10,797	0.279	1,111	0.320
non-linearity + non-normality	8,835	0.472	928	0.600
heteroskedasticity + non-normality	7,870	0.354	819	0.489
non-linearity + heteroskedasticity + non-normality	7,721	0.432	717	0.620

tributions in both predictors and error terms. A comprehensive grid of parameter combinations was explored to generate a wide range of residual patterns. Importantly, the simulation included scenarios with multiple violations occurring simultaneously. To ensure uniform coverage across the difficulty scale of the target variable D, a bucket sampling scheme was used to create a balanced dataset. Samples were iteratively simulated and accepted into one of 50 buckets, each representing a distinct range of D. Table 1 summarizes the number of training and test samples under the different violation scenarios.

The computer vision model was trained on paired data consisting of an RGB residual plot image as input and the corresponding target value D, with the objective of learning to estimate $\hat{D}$. Model training was conducted on the MASSIVE M3 high-performance computing platform (Goscinski et al. 2014) using TensorFlow (Abadi et al. 2016) and Keras (Chollet et al. 2015). Hyperparameters were optimized via Bayesian tuning with KerasTuner (O'Malley et al. 2019), minimizing validation root mean square error (RMSE) across 100 trials. The tuning process included early stopping and considered dropout rate, batch normalization, input resolution, and auxiliary inputs.

Further details, including mathematical formulations of the synthetic data model, parameter specifications, sampling scheme, example residual plots and hyperparameter tuning configuration, are provided in the Appendix A and C.

8. Results

8.1. *Model Performance*

Table 2 summarizes the test performance of optimized models using three input sizes. The 32×32 model consistently outperformed others, with a mean absolute error of approximately 0.43, a negligible deviation given the typical D range of 0 to 7. High R^2 values also indicated strong linear correlation between predictions and targets. Based on these results, we used the best-performing 32×32 model for subsequent analyses.

Figure 4 shows a hexagonal heatmap of $D - \hat{D}$ versus D. Smoothing curves (fitted by generalized additive models, Hastie 2017) reveal that all models performed well for $1.5 < D < 6$, where no structural issues were observed. Over-prediction occurred when $D < 1.5$, and under-prediction primarily when $\hat{D} > 6$.

For null plots ($D = 0$), over-prediction was expected (see Section 5.1), but not under-prediction. Therefore, we analysed the relationship between residuals and all the factors involved in the data generating process. We found that most deviations stemmed from non-linearity and the presence of a second predictor in the synthetic data model, where small ε variances caused under-predictions and large ε variances caused over-prediction, as illustrated in Figure 5.

To investigate further, we stratified the train and test set by violation type and re-evaluated performance (Table 1). It was found that null plots had the poorest metrics, while non-normality cases were easiest to predict, with RMSE around 0.3. Non-linearity (RMSE ≈ 0.8) was harder to assess than heteroskedasticity (RMSE ≈ 0.6) or non-normality (RMSE ≈ 0.3). Plots with multiple violations showed intermediate performance.

8.2. *Comparison with Human Visual Inference and Conventional Tests*

8.2.1. Overview of the Human Subject Experiment

In order to check the validity of the proposed computer vision model, residual plots presented in the human subject experiment conducted by Li et al. (2024) will be assessed.

Table 2. The test performance of three optimized models with different input sizes. The metrics are computed by comparing the estimated distance $\hat{D}$ (model output) and the target distance D generated from the synthetic data model. RMSE represents the root mean squared error; R^2 is the squared correlation between the two quantities; MAE denotes the mean absolute error; and Huber loss refers to the average Huber loss computed over the test set.

	RMSE	R^2	MAE	Huber loss
32×32	0.660	0.901	0.434	0.18
64×64	0.674	0.897	0.438	0.19
128×128	0.692	0.892	0.460	0.20

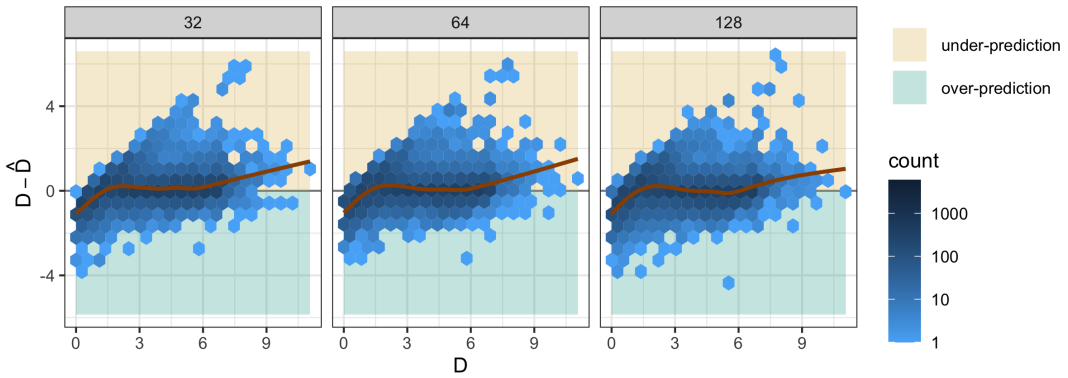

Figure 4. Hexagonal heatmap for difference in D and $\hat{D}$ vs D on test data for three optimized models with different input sizes. The brown lines are smoothing curves produced by fitting generalized additive models. Over-prediction and under-prediction can be observed for small D and large D respectively.

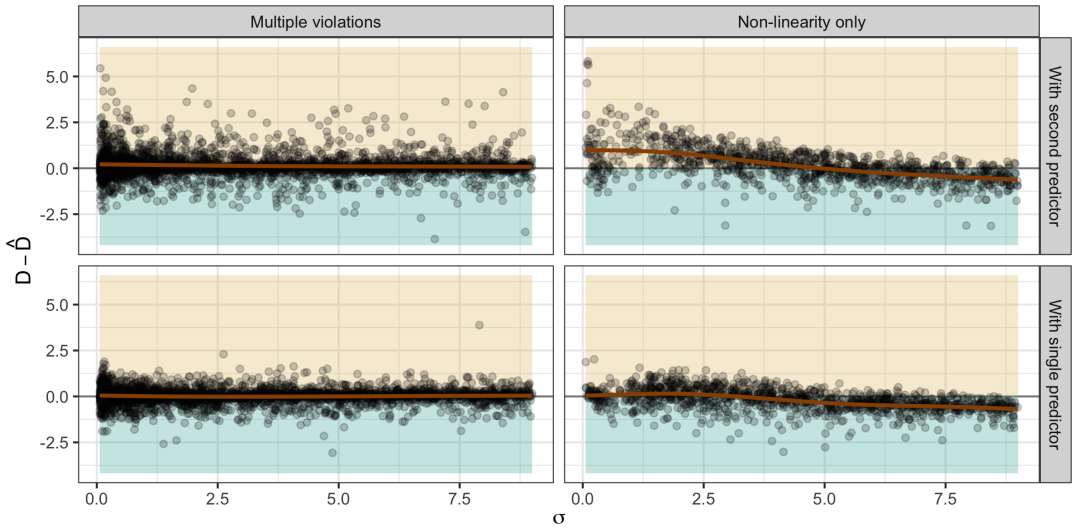

Figure 5. Scatter plots for difference in D and $\hat{D}$ vs σ on test data for the 32×32 optimized model. The data is grouped by whether the regression has only non-linearity violation, and whether it includes a second predictor in the regression formula. The brown lines are smoothing curves produced by fitting generalized additive models. Under-prediction mainly occurs when the data-generating process has small σ, a second predictor, and only non-linearity as the model violation.

This study has collected 7,974 human responses to 1,152 lineups. Each lineup contains one randomly placed true residual plot and 19 null plots. Among the 1,152 lineups, 24 are attention check lineups in which the visual patterns are designed to be extremely obvious and very different from the corresponding to null plots, 36 are null lineups where all the lineups consist of only null plots, 279 are lineups with uniform predictor distribution evaluated by 11 participants, and the remaining 813 are lineups with discrete, skewed or normal predictor distribution evaluated by 5 participants. Attention check lineups and null lineups will not be assessed in the following analysis.

In Li et al. (2024), the residual plots are simulated from a data generating process that corresponds to a special case of the synthetic data model used in this study. A key feature of the design is that model violations are introduced independently, meaning non-linearity and heteroskedasticity do not co-exist within a single lineup but are instead assigned uniformly across different lineups. Moreover, the experimental design does not account for non-normality or multiple predictors.

8.2.2. Model Performance on the Human-evaluated Data

Each lineup in Li et al. (2024) contains one true residual plot and 19 null plots. While the true residual plot's distance D depends on the data-generating process, null plots have $D = 0$. We used our optimized computer vision model to estimate D for all plots. To ensure fair comparison, we reject H_0 if the true residual plot has the highest estimated distance. Conventional tests, including the Ramsey RESET (Ramsey 1969) for non-linearity and the Breusch-Pagan test (Breusch and Pagan 1979) for heteroskedasticity, were also applied to the same data.

Table 3 reports performance metrics for $\hat{D}$ on true residual plots. While the performance is slightly worse than on the test data provided in Table 2, the mean absolute error remains low and the correlation between predictions and true values remains high. Consistent with results in Table 1, non-linearity cases are harder to predict than heteroskedasticity cases.

Table 4 compares the model's rejection decisions with those of conventional tests and human visual tests. Agreement rates are 85.95% for heteroskedasticity cases and 79.69% for non-linearity cases, higher than those observed for human visual tests

Table 3. The performance of the 32×32 model on the data used in the human subject experiment.

Violation	RMSE	R^2	MAE	Huber loss
heteroskedasticity	0.721	0.852	0.553	0.235
non-linearity	0.738	0.770	0.566	0.246

Table 4. Summary of the comparison of decisions made by computer vision model with decisions made by conventional tests and visual tests conducted by human.

Violations	#Samples	#Agreements	Agreement rate
Compared with conventional tests			
heteroskedasticity	540	464	0.8593
non-linearity	576	459	0.7969
Compared with visual tests conducted by human			
heteroskedasticity	540	367	0.6796
non-linearity	576	385	0.6684

(67.96% and 66.84%, respectively). This suggests that the model aligns more closely with the best available conventional tests. However, as Figure 6 shows, the model does not always reject when conventional tests do, indicating a lower sensitivity to model violations.

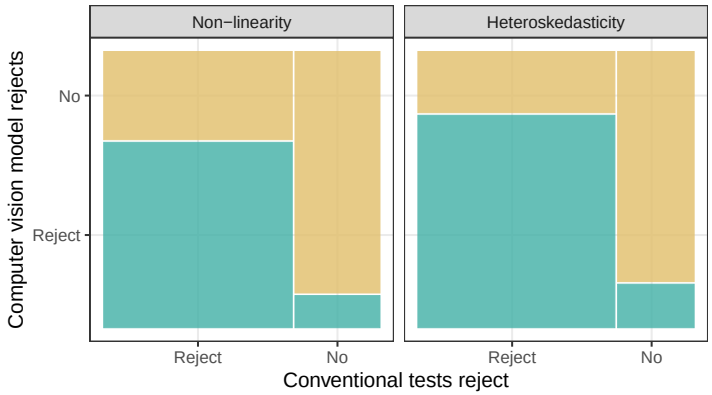

Figure 6. Rejection rate (p-value ≤ 0.05) of computer vision models conditional on conventional tests on non-linearity (left) and heteroskedasticity (right) lineups displayed using a mosaic plot. When the conventional test fails to reject, the computer vision mostly fails to reject the same plot as well as indicated by the height of the top right yellow rectangle, but there are non negliable amount of plots where the conventional test rejects but the computer vision model fails to reject as indicated by the width of the top left yellow rectangle.

To further illustrate these patterns, Figure 7 presents parallel coordinate plots comparing decisions from human visual tests, the computer vision model, and conventional tests. All three agree in about 50% of cases. When humans do not reject, conventional

tests reject in about half of those cases, whereas the model rejects in only 25%. Notably, the model rarely rejects when both humans and conventional tests do not, suggesting that while it is more sensitive than humans, its rejection behavior is more aligned with human judgment than with conventional tests.

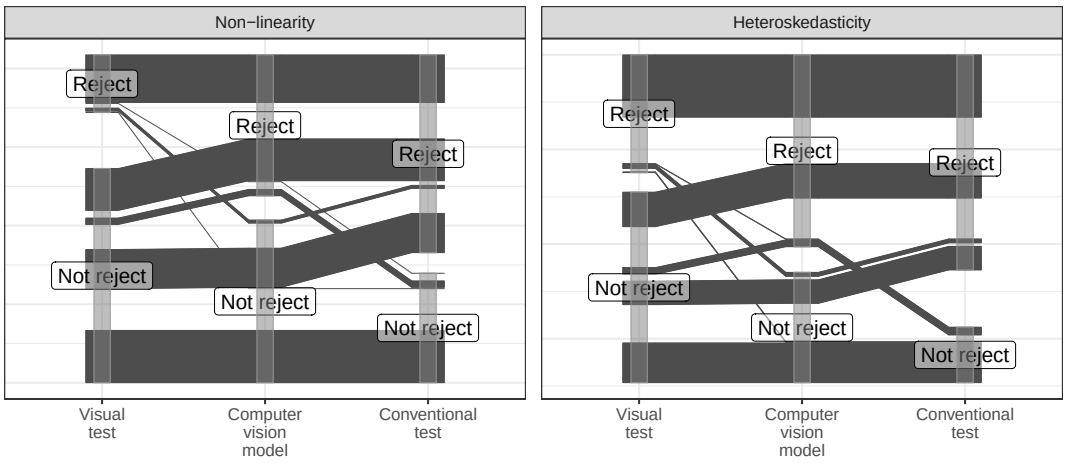

Figure 7. Parallel coordinate plots of decisions made by computer vision model, conventional tests and visual tests made by human. All three agree in around 50% of cases. The model rejects less often than conventional tests when humans do not, and rarely rejects when both others do not, indicating closer alignment with human judgment.

Figure 8 plots rejection decisions against the true distance D. Compared to conventional tests, the computer vision model rejects fewer cases when $D < 4$, but a similar number of cases when $D > 4$. This indicates that the model is less sensitive to minor deviations from model assumptions, while remaining comparably sensitive to more substantial violations. Visual tests are the least sensitive overall, showing few rejections for $D < 2$, and requiring a higher threshold ($D > 5$) for near-universal rejection, compared to $D > 4$ for both the model and conventional tests.

Finally, the power curves in Figure 8 are fitted using logistic regression without intercepts and with an offset of $\log(0.05/0.95)$ to ensure a 5% rejection rate under H_0. The model's rejection curve consistently lies between those of conventional and human visual tests, indicating potential to further align its decision boundary with human visual assessment.

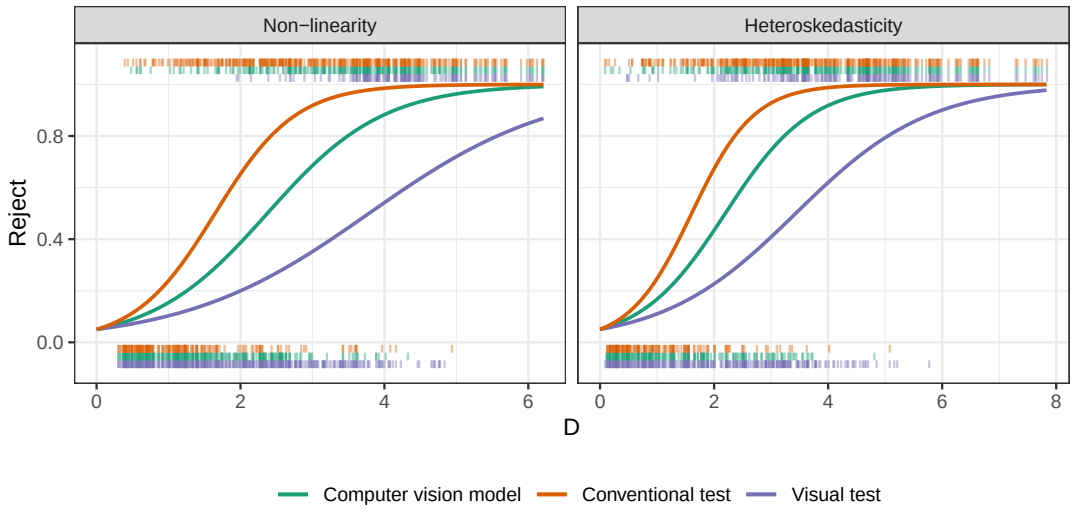

Figure 8. Comparison of power of visual tests, conventional tests and the computer vision model. Marks along the x-axis at the bottom of the plot represent rejections made by each type of test. Marks at the top of the plot represent acceptances. Power curves are fitted by logistic regression models with no intercept but an offset equals to log(0.05/0.95). The model is less sensitive than conventional tests for small D but similarly sensitive for large D. Visual tests are least sensitive overall. The model's curve lies between those of conventional and visual tests.

8.2.3. Adjusted δ-difference

In the experiment conducted by Li et al. (2024), participants could select multiple plots per lineup. To reflect detection confidence, a weighted detection rate was computed for each lineup: selecting zero plots yielded a weight of 0.05; otherwise, if the true residual plot was selected, its weight was the inverse of the number of selections.

To evaluate the proposed distance measure, we could examine the relationship between the weighted detection rate and the δ-difference statistic from Chowdhury et al. (2018):

$$\delta = \bar{d}_{\text{true}} - \max_j \left(\bar{d}^{(j)}_{\text{null}} \right) \quad \text{for } j = 1, ..., m-1,$$

where $\bar{d}_{\text{true}}$ is the mean distance between the true residual plot and all null plots, and $\bar{d}^{(j)}_{\text{null}}$ is the mean distance from null plot j to other nulls. These averages help address asymmetries in pairwise distances.

However, our method only compares each plot to a theoretically "good" residual plot, making the original formulation inapplicable. Moreover, since $D_{\text{null}} = 0$ by def-

inition and D can not be derived from an image, we instead focus on $\hat{D}$, the model-estimated distance. This leads to the adjusted δ-difference:

$$\delta_{\text{adj}} = \hat{D} - \max_j \left(\hat{D}^{(j)}_{\text{null}} \right) \quad \text{for } j = 1, ..., m-1,$$

where $\hat{D}^{(j)}_{\text{null}}$ is the estimated distance for the j-th null plot, and m is the number of plots in a lineup.

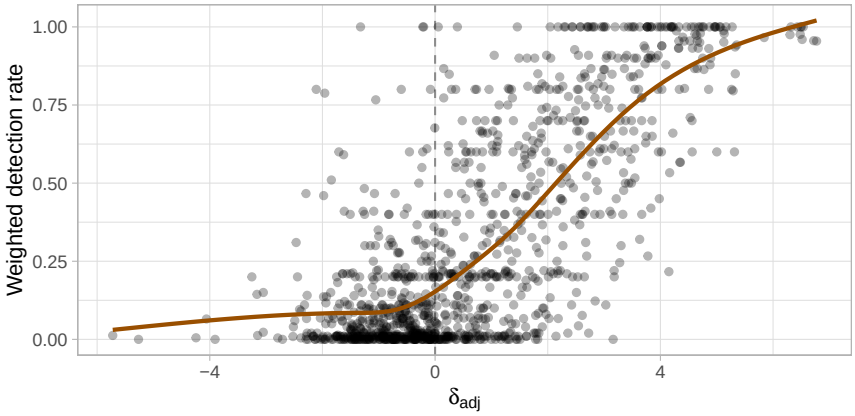

Figure 9. A weighted detection rate vs adjusted δ-difference plot. The brown line is smoothing curve produced by fitting generalized additive models. Detection generally increases with positive δ_{adj}, but variability and exceptions highlight the distance measure's imperfect alignment with human perception.

Figure 9 displays the scatter plot of the weighted detection rate against δ_{adj}. It shows that weighted detection generally increases with δ_{adj}, especially when it is positive. This suggests that the differences in $\hat{D}$ capture some aspect of the visual separability between the true and null plots as perceived by humans, with larger positive differences generally indicating greater distinctiveness. However, the considerable variability in detection rates for similar δ_{adj} values implies that $\hat{D}$ does not fully account for all factors influencing human perception, and the alignment remains imperfect. A negative δ_{adj} suggests some null plots may appear more visually distinctive to humans than the true residual plot. In some instances, the weighted detection rate is close to one despite a negative δ_{adj}. This discrepancy implies that the distance measure may not perfectly reflect actual human behavior.

9. Examples

In this section, we present the performance of the trained computer vision model on three example datasets, including the dataset associated with the residual plot displaying a "left-triangle" shape, the Boston housing dataset (Harrison Jr and Rubinfeld 1978), and the "dino" datasets from the from the `datasauRus` package (Davies, Locke, and D'Agostino McGowan 2022).

In the first case, both the model and human inspection correctly avoid rejecting H_0 when it is true, contrary to conventional tests, highlighting the necessity of visual examination. The second case shows a clear model violation detected by all tests, demonstrating the model's practical use for less intricate tasks. The third case involves an unusual deviation where the model and humans reject H_0, but some conventional tests do not, illustrating the benefits of visual-based decision-making.

9.1. *Left-triangle*

In Section 1, we introduced a residual plot (Figure 1) where the "left-triangle" shape could be misinterpreted as evidence of heteroskedasticity. Although the residuals are drawn from a correctly specified model, the Breusch-Pagan test rejects H_0 ($p = 0.046$). This visual artifact stems from the skewed distribution of fitted values, as illustrated in the lineup in Figure 13A, where similar "left-triangle" patterns appear across all residual plots. Since the true plot (at position 10) is visually indistinguishable from the null plots, a visual test would not reject H_0.

The computer vision model yields a consistent conclusion. As shown in Figure 10, the estimated distance $\hat{D}$ falls well below the 95% quantile of the null distribution, with a p-value of 0.33. The bootstrapped distribution further suggests that rejection of the fitted model is highly unlikely. Together, the visual and model-based tests do not reject H_0, in contrast to the Breusch-Pagan test, which lacks the context provided by null plots and may be misled by visual artifacts.

To further interpret the model's output, the attention map in Figure 10B, computed as the gradient of the output with respect to the grayscale input, shows that the top-right and bottom-right corners of the residual plot have the most influence,

corresponding to the triangle's vertices. In addition, a principal component analysis (PCA) of the 256 features extracted by the model's global pooling layer (Figure 10D) shows that null and bootstrapped plots cluster together in the space defined by the first two principal components, and the true residual plot is covered by the cluster. This overlap reinforces the conclusion that the true plot is visually indistinct, aligning with our interpretation of the lineup.

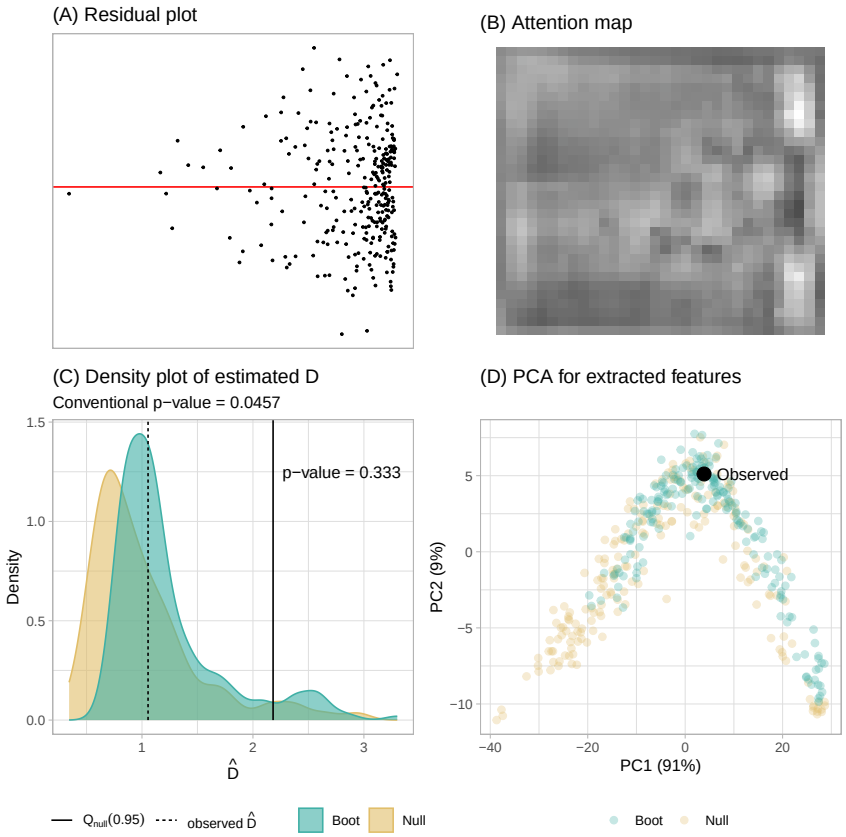

Figure 10. A summary of the residual plot assessment evaluated on 200 null plots and 200 bootstrapped plots. (A) The true residual plot exhibiting a "left-triangle" shape. (B) The attention map highlights the top-right and bottom-right corners of the residual plot as the most influential. (C) The density plot shows estimated distances for null (yellow) and bootstrapped (green) plots. The fitted model is not rejected because $\hat{D} < Q_{null}(0.95)$. (D) Null and bootstrapped plots cluster together in the space defined by the first two principal components of the global pooling layer. The cluster also covers the true residual plot.

9.2. *Boston Housing*

The Boston housing dataset, originally published by Harrison Jr and Rubinfeld (1978), provides insights into housing in the Boston, Massachusetts area. For illustration, we use a reduced Kaggle version containing 489 observations and four variables: average

number of rooms per dwelling (RM), percentage of lower status population (LSTAT), pupil-teacher ratio by town (PTRATIO), and median home value (MEDV). We fit a linear regression model with MEDV as the response and the remaining variables as predictors, focusing on detecting non-linearity, because the relationships between RM and MEDV or LSTAT and MEDV are non-linear.

Figure 11 summarizes the model evaluation. The residual plot (Figure 11A) reveals a distinct "U"-shaped pattern, suggesting non-linearity. The RESET test strongly rejects H_0 with a very small p-value, and the computer vision model yields a large estimated distance $\hat{D}$, well above the 95% quantile of the null distribution (p-value = 0.00498). The bootstrapped distribution also shows widespread rejection, indicating the model is likely misspecified. The attention map (Figure 11B) highlights the central region of the residual plot, corresponding to the turning point of the "U", as most influential. In the PCA projection (Figure 11D), bootstrapped and null plots form two distinct clusters, emphasizing the visual separability. This aligns with the lineup in Figure 13B, where the true plot stands out clearly. A human visual test would also likely reject H_0 in this case.

9.3. *DatasauRus*

Beyond detecting common issues like non-linearity, heteroskedasticity, and non-normality, the computer vision model can also identify unusual visual artifacts, such as shapes resembling real-world objects, as long as such patterns do not appear in null plots. These cases are challenging to categorize using conventional model diagnostics, so we compare the model's output with results from the RESET test, Breusch-Pagan test, and Shapiro-Wilk test (Shapiro and Wilk 1965).

The "dino" dataset from the `datasauRus` R package illustrates this situation. It contains only two variables, x and y, but fitting a regression model produces a residual plot that clearly resembles a dinosaur (Figure 12A). This pattern is visually distinctive in the lineup shown in Figure 13C, where a human visual test would undoubtedly reject H_0.

Consistent with this, the computer vision model assigns a distance $\hat{D}$ above the 95% quantile of the null distribution (p-value = 0.00498), leading to rejection of H_0. The

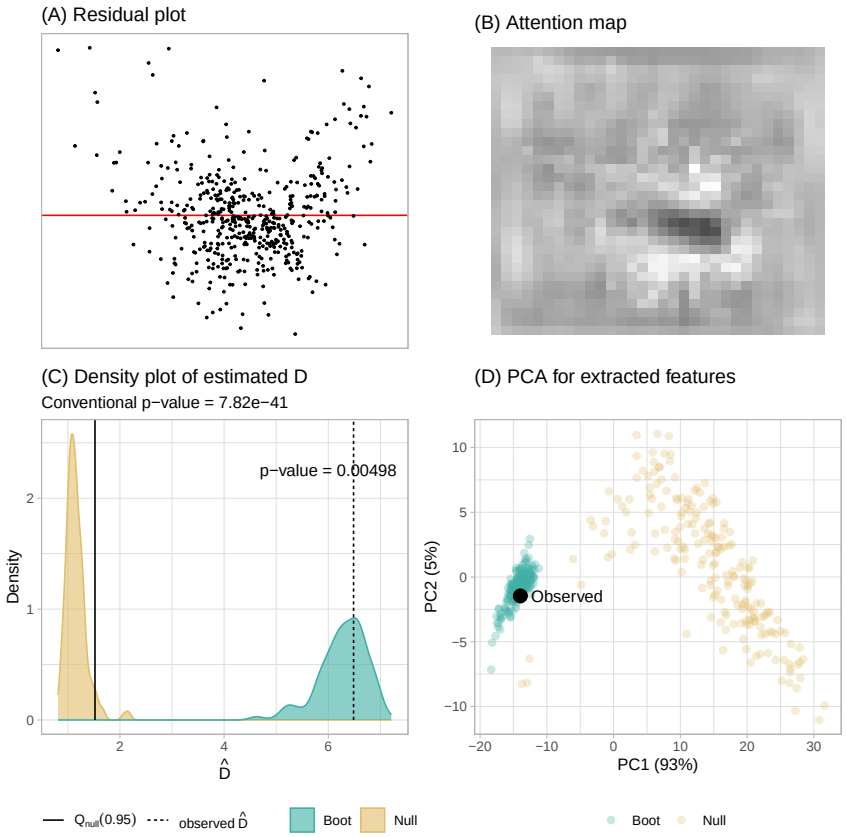

Figure 11. A summary of the residual plot assessment for the Boston housing fitted model evaluated on 200 null plots and 200 bootstrapped plots. (A) The true residual plot exhibiting a "U" shape. (B) The attention map highlights the central region of the "U" shape as the most influential part of the residual plot. (C) The density plot shows estimated distances for null (yellow) and bootstrapped (green) plots. The fitted model is rejected because $\hat{D} \geq Q_{null}(0.95)$. (D) Bootstrapped and null plots form distinct clusters in the space of the first two principal components from the global pooling layer, highlighting their visual separability.

bootstrapped distribution further supports this, with most fitted models also being rejected. In contrast, the RESET and Breusch-Pagan tests yield p-values above 0.3, suggesting no violation. Only the Shapiro-Wilk test rejects normality with a small p-value.

Crucially, the attention map in Figure 12B highlights the dinosaur shape, indicating that the model's decision is driven by human-perceptible features. The model appears to capture contours and outlines in a manner similar to human visual interpretation. In the PCA plot (Figure 12D), the cluster of bootstrapped plots is positioned at the corner of the cluster of null plots, isolated from the cluster of null plots.

In practice, such visual artifacts would be difficult to detect without inspecting the residual plot directly. Furthermore, analysts do not always test for normality, and

even when violations are found, they are often deemed acceptable due to the robustness of linear regression under quasi-likelihood theory. This example highlights the value of visual diagnostics and the unique role of the proposed computer vision model in supporting them.

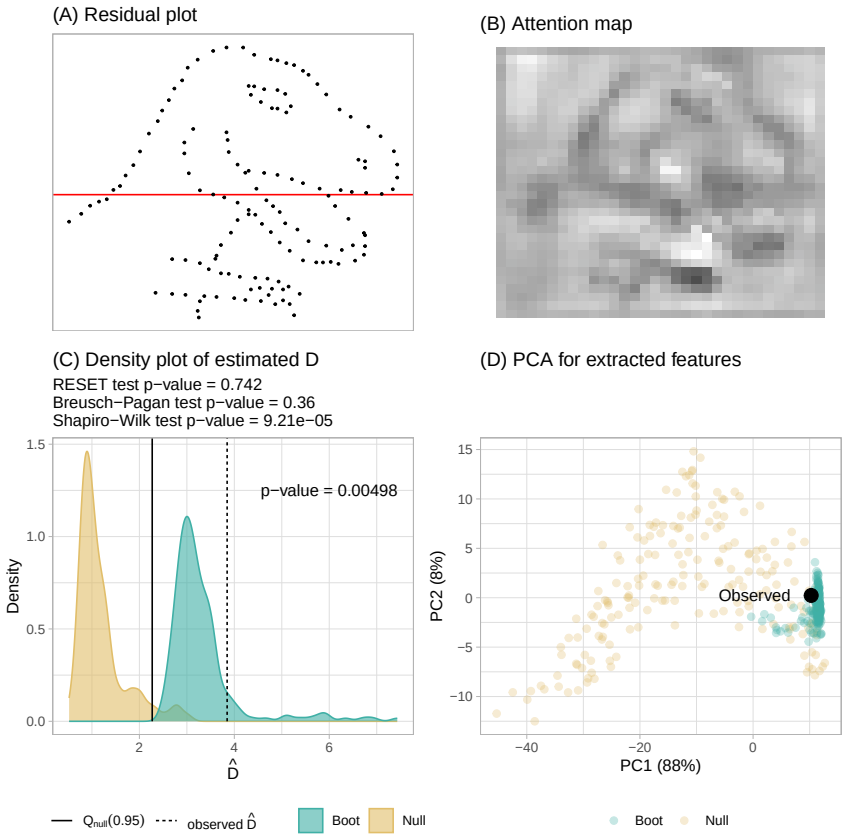

Figure 12. A summary of the residual plot assessment for the datasauRus fitted model evaluated on 200 null plots and 200 bootstrapped plots. (A) The residual plot exhibits a "dinosaur" shape. (B) The attention map highlights the dinosaur shape, indicating the model's decision is driven by human-recognizable features.(C) The density plot shows estimated distances for null (yellow) and bootstrapped (green) plots. The fitted model is rejected because $\hat{D} \geq Q_{null}(0.95)$. (D) The bootstrapped plots cluster at the corner of the null plot cluster, yet remain isolated in the space defined by the first two principal components of the global pooling layer.

10. Limitations and Future Work

While the computer vision model performs well under the synthetic data generation scheme and across the three illustrative examples, this study has several limitations that point to directions for future work.

First, the proposed distance measure assumes the true model follows a classical nor-

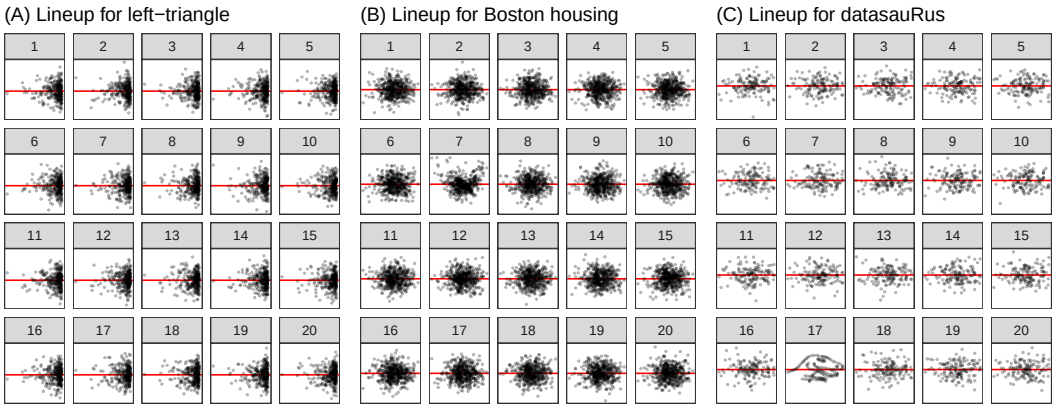

Figure 13. Lineups of residual plots for the "left-triangle", "Boston housing", and "datasauRus" datasets. (A) True plot at position 10; no clear visual difference. (B) True plot at position 7; "U"-shape clearly stands out. (C) True plot at position 17; distinctly artificial and easily identifiable.

mal linear regression framework, which may be restrictive. Although we do not explore relaxation of this assumption here, future work could extend the approach to other types of regression models. One possibility is to define a distance measure tailored to each model class and re-train the computer vision model accordingly. To reduce computational cost, the convolutional blocks from our current model could be reused via transfer learning, as it is already effective at capturing visual features. Alternatively, transforming residuals to approximate normality and constant variance could allow for meaningful comparisons under our current distance measure. However, if raw residuals are used, the interpretation remains valid only when the visual differences between the true and null plots are detectable by the proposed distance measure.

Second, this study focused solely on the standard residuals vs. fitted values plot, leaving out other common diagnostic plots such as residuals vs. predictors and Q-Q plots. These alternative plots may offer additional information and could be incorporated into a broader visual testing framework in future research.

Finally, we chose a relatively simple computer vision architecture to establish a baseline for automated visual inference. While the current model shows promise, there is room to improve its alignment with human judgment. Achieving this may require external data from surveys or controlled experiments to better understand how human observers interpret residual plots and how these judgments differ from the model's outputs.

11. Conclusions

In this paper, we have introduced a distance measure based on the Kullback-Leibler divergence to quantify disparity between the residual distribution of a fitted classical normal linear regression model and the reference distribution under correct specification. This measure captures the extent of model violations and forms the basis of a computer vision model that estimates the distance using residual plots as input.

The estimated distance supports a formal statistical testing procedure by comparing the true residual plot against null plots generated from the fitted model. Through bootstrapping and model refitting, we also evaluate how often a correctly specified model would be flagged as misspecified under repeated sampling.

The trained computer vision model performs well on both training and test data, though its performance is somewhat lower for non-linear patterns. Statistical tests based on the model's predicted distance show lower sensitivity than conventional tests but higher sensitivity than human visual tests. Importantly, the predicted distance broadly aligns with the strength of visual signals perceived by humans, though further improvements are possible.

We demonstrate the method across several scenarios, highlighting its effectiveness and its conceptual alignment with visual inference. Both visual and distance-based tests act as omnibus procedures, detecting a wide range of model violations collectively. In contrast, conventional tests such as RESET and Breusch-Pagan target specific assumptions, making them less flexible and potentially harder to combine without adjusting for multiple testing.

Overall, our approach offers a practical tool to support regression diagnostics by reducing the manual effort required for residual plot inspection. While we recommend analysts to continue reading residual plots whenever feasible, our method can augment the process, either as an automated screening tool or a supplement to human judgment.

Supplementary Materials

Appendix: The appendix includes more details about the synthetic data generation process, neural network layers used in the work, model training, and model

violation index. (appendix.pdf, Portable Document Format file)

Acknowledgement

These `R` packages were used for the work: `tidyverse` (Wickham et al. 2019), `lmtest` (Zeileis and Hothorn 2002), `mpoly` (Kahle 2013), `ggmosaic` (Jeppson, Hofmann, and Cook 2021), `kableExtra` (Zhu 2021), `patchwork` (Pedersen 2022), `glue` (Hester and Bryan 2022), `ggpcp` (Hofmann, VanderPlas, and Ge 2022), `here` (Müller 2020), and `reticulate` (Ushey, Allaire, and Tang 2024).

The article was created with R packages `rticles` (Allaire et al. 2022), `knitr` (Xie 2014) and `rmarkdown` (Xie, Dervieux, and Riederer 2020). The project's GitHub repository (`https://github.com/TengMCing/auto_residual_reading_paper`) contains all materials required to reproduce this article.

We acknowledge the use of ChatGPT (OpenAI, GPT-4, accessed July 2025) to improve the clarity and fluency of the manuscript text. All content was reviewed and verified by the authors to ensure accuracy and integrity.

Data Availability Statement

The data that support the findings of this study are openly available in the project's GitHub repository (`https://github.com/TengMCing/auto_residual_reading_paper`).

Disclosure Statement

No potential conflict of interest was reported by the author(s).

References

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, et al. 2016. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* .

Allaire, JJ, Yihui Xie, Christophe Dervieux, R Foundation, Hadley Wickham, Journal of Statistical Software, Ramnath Vaidyanathan, et al. 2022. *rticles: Article formats for R Markdown*. R package version 0.24, `https://CRAN.R-project.org/package=rticles`.

Belsley, David A, Edwin Kuh, and Roy E Welsch. 1980. *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley & Sons.

Breusch, Trevor S, and Adrian R Pagan. 1979. "A simple test for heteroscedasticity and random coefficient variation." *Econometrica: Journal of the Econometric Society* 1287–1294.

Brunetti, Antonio, Domenico Buongiorno, Gianpaolo Francesco Trotta, and Vitoantonio Bevilacqua. 2018. "Computer vision and deep learning techniques for pedestrian detection and tracking: A survey." *Neurocomputing* 300: 17–33.

Buja, Andreas, Dianne Cook, Heike Hofmann, Michael Lawrence, Eun-Kyung Lee, Deborah F Swayne, and Hadley Wickham. 2009. "Statistical inference for exploratory data analysis and model diagnostics." *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 367 (1906): 4361–4383.

Chen, Yun, Shijie Su, and Hui Yang. 2020. "Convolutional neural network analysis of recurrence plots for anomaly detection." *International Journal of Bifurcation and Chaos* 30 (01): 2050002.

Chollet, François, et al. 2015. "Keras." `https://keras.io`.

Chopra, Sumit, Raia Hadsell, and Yann LeCun. 2005. "Learning a similarity metric discriminatively, with application to face verification." In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Vol. 1, 539–546. IEEE.

Chowdhury, Niladri Roy, Dianne Cook, Heike Hofmann, and Mahbubul Majumder. 2018. "Measuring lineup difficulty by matching distance metrics with subject choices in crowdsourced data." *Journal of Computational and Graphical Statistics* 27 (1): 132–145.

Chu, Hongyang, Xinwei Liao, Peng Dong, Zhiming Chen, Xiaoliang Zhao, and Jiandong Zou. 2019. "An automatic classification method of well testing plot based on convolutional neural network (CNN)." *Energies* 12 (15): 2846.

Cook, R Dennis, and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Davies, Rhian, Steph Locke, and Lucy D'Agostino McGowan. 2022. *datasauRus: Datasets from the Datasaurus Dozen*. R package version 0.1.6, `https://CRAN.R-project.org/package=datasauRus`.

Davison, Anthony Christopher, and David Victor Hinkley. 1997. *Bootstrap methods and their*

application. Cambridge university press.

Efron, Bradley, and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. Chapman and Hall/CRC.

Emami, Shervin, and Valentin Petrut Suciu. 2012. "Facial recognition using OpenCV." *Journal of Mobile, Embedded and Distributed Systems* 4 (1): 38–43.

Fieberg, John, Smith Freeman, and Johannes Signer. 2024. "Using lineups to evaluate goodness of fit of animal movement models." *Methods in Ecology and Evolution* .

Frisch, Ragnar, and Frederick V Waugh. 1933. "Partial time regressions as compared with individual trends." *Econometrica: Journal of the Econometric Society* 387–401.

Fukushima, Kunihiko, and Sei Miyake. 1982. "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position." *Pattern recognition* 15 (6): 455–469.

Gebhardt, Albrecht, Roger Bivand, and David Sinclair. 2023. *interp: Interpolation Methods*. R package version 1.1-5, `https://CRAN.R-project.org/package=interp`.

Glorot, Xavier, and Yoshua Bengio. 2010. "Understanding the difficulty of training deep feedforward neural networks." In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256. JMLR Workshop and Conference Proceedings.

Goscinski, Wojtek J, Paul McIntosh, Ulrich Felzmann, Anton Maksimenko, Christopher J Hall, Timur Gureyev, Darren Thompson, et al. 2014. "The multi-modal Australian ScienceS Imaging and Visualization Environment (MASSIVE) high performance computing infrastructure: applications in neuroscience and neuroinformatics research." *Frontiers in Neuroinformatics* 8: 30.

Hailesilassie, Tameru. 2019. "Financial Market Prediction Using Recurrence Plot and Convolutional Neural Network." .

Harrison Jr, David, and Daniel L Rubinfeld. 1978. "Hedonic housing prices and the demand for clean air." *Journal of environmental economics and management* 5 (1): 81–102.

Hastie, Trevor J. 2017. "Generalized additive models." In *Statistical models in S*, 249–307. Routledge.

Hatami, Nima, Yann Gavet, and Johan Debayle. 2018. "Classification of time-series images using deep convolutional neural networks." In *Tenth international conference on machine vision (ICMV 2017)*, Vol. 10696, 242–249. SPIE.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern*

recognition, 770–778.

Hermite, M. 1864. *Sur un nouveau développement en série des fonctions*. Imprimerie de Gauthier-Villars.

Hester, Jim, and Jennifer Bryan. 2022. *glue: Interpreted String Literals*. R package version 1.6.2, `https://CRAN.R-project.org/package=glue`.

Hofmann, Heike, Susan VanderPlas, and Yawei Ge. 2022. *ggpcp: Parallel Coordinate Plots in the 'ggplot2' Framework*. R package version 0.2.0, `https://CRAN.R-project.org/package=ggpcp`.

Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. "Densely connected convolutional networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708.

Hyndman, Rob J, and Yanan Fan. 1996. "Sample quantiles in statistical packages." *The American Statistician* 50 (4): 361–365.

Jeppson, Haley, Heike Hofmann, and Di Cook. 2021. *ggmosaic: Mosaic plots in the 'ggplot2' framework*. R package version 0.3.3, `https://CRAN.R-project.org/package=ggmosaic`.

Kahle, David. 2013. "mpoly: Multivariate Polynomials in R." *The R Journal* 5 (1): 162–170.

Kingma, Diederik P, and Jimmy Ba. 2014. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* .

Krishnan, Ganesh, and Heike Hofmann. 2021. "Hierarchical Decision Ensembles-An inferential framework for uncertain Human-AI collaboration in forensic examinations." *arXiv preprint arXiv:2111.01131* .

Kullback, Solomon, and Richard A Leibler. 1951. "On information and sufficiency." *The Annals of Mathematical Statistics* 22 (1): 79–86.

Langsrud, Øyvind. 2005. "Rotation tests." *Statistics and computing* 15: 53–60.

Lee, Howard, and Yi-Ping Phoebe Chen. 2015. "Image based computer aided diagnosis system for cancer detection." *Expert Systems with Applications* 42 (12): 5356–5365.

Li, Weihao, Dianne Cook, Emi Tanaka, and Susan VanderPlas. 2024. "A plot is worth a thousand tests: Assessing residual diagnostics with the lineup protocol." *Journal of Computational and Graphical Statistics* 1–19.

Loy, Adam, and Heike Hofmann. 2013. "Diagnostic tools for hierarchical linear models." *Wiley Interdisciplinary Reviews: Computational Statistics* 5 (1): 48–61.

Loy, Adam, and Heike Hofmann. 2014. "HLMdiag: A suite of diagnostics for hierarchical linear models in R." *Journal of Statistical Software* 56: 1–28.

Loy, Adam, and Heike Hofmann. 2015. "Are you normal? The problem of confounded residual structures in hierarchical linear models." *Journal of Computational and Graphical Statistics* 24 (4): 1191–1209.

Mason, Harriet, Stuart Lee, Ursula Laa, and Dianne Cook. 2022. *cassowaryr: Compute Scagnostics on Pairs of Numeric Variables in a Data Set*. R package version 2.0.0, `https://CRAN.R-project.org/package=cassowary`.

Müller, Kirill. 2020. *here: A simpler way to find your files*. R package version 1.0.1, `https://CRAN.R-project.org/package=here`.

Nikolenko, Sergey I, et al. 2021. *Synthetic data for deep learning*. Vol. 174. Springer.

Ojeda, Sun Arthur A, Geoffrey A Solano, and Elmer C Peramo. 2020. "Multivariate time series imaging for short-term precipitation forecasting using convolutional neural networks." In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 33–38. IEEE.

O'Malley, Tom, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. 2019. "Keras Tuner." `https://github.com/keras-team/keras-tuner`.

Pedersen, Thomas Lin. 2022. *patchwork: The composer of plots*. R package version 1.1.2, `https://CRAN.R-project.org/package=patchwork`.

Ramsey, James Bernard. 1969. "Tests for specification errors in classical linear least-squares regression analysis." *Journal of the Royal Statistical Society: Series B (Methodological)* 31 (2): 350–371.

Rawat, Waseem, and Zenghui Wang. 2017. "Deep convolutional neural networks for image classification: A comprehensive review." *Neural computation* 29 (9): 2352–2449.

Series, BT. 2011. "Studio encoding parameters of digital television for standard 4: 3 and widescreen 16: 9 aspect ratios." *International Telecommunication Union, Radiocommunication Sector* .

Shapiro, Samuel Sanford, and Martin B Wilk. 1965. "An analysis of variance test for normality (complete samples)." *Biometrika* 52 (3/4): 591–611.

Silverman, Bernard W. 2018. *Density estimation for statistics and data analysis*. Routledge.

Simonyan, Karen, and Andrew Zisserman. 2014. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* .

Singh, Karamjit, Garima Gupta, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2017. "Deep convolutional neural networks for pairwise causality." *arXiv preprint arXiv:1701.00597* .

Tukey, John W, and Paul A Tukey. 1985. "Computer graphics and exploratory data analysis: An introduction." In *Proceedings of the sixth annual conference and exposition: computer graphics*, Vol. 85, 773–785.

Ushey, Kevin, JJ Allaire, and Yuan Tang. 2024. *reticulate: Interface to 'Python'*. R package version 1.35.0, `https://CRAN.R-project.org/package=reticulate`.

Wang, Zhou, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. "Image quality assessment: from error visibility to structural similarity." *IEEE transactions on image processing* 13 (4): 600–612.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686.

Widen, Holly M, James B Elsner, Stephanie Pau, and Christopher K Uejio. 2016. "Graphical inference in geographical research." *Geographical Analysis* 48 (2): 115–131.

Wilkinson, Leland, Anushka Anand, and Robert Grossman. 2005. "Graph-theoretic scagnostics." In *Information Visualization, IEEE Symposium on*, 21–21. IEEE Computer Society.

Xie, Yihui. 2014. "knitr: A comprehensive tool for reproducible research in R." In *Implementing reproducible computational research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman and Hall/CRC. ISBN 978-1466561595, `http://www.crcpress.com/product/isbn/9781466561595`.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown cookbook*. Boca Raton, Florida: Chapman and Hall/CRC. ISBN 9780367563837, `https://bookdown.org/yihui/rmarkdown-cookbook`.

Zeileis, Achim, and Torsten Hothorn. 2002. "Diagnostic checking in regression relationships." *R News* 2 (3): 7–10.

Zhang, Ye, Yi Hou, Shilin Zhou, and Kewei Ouyang. 2020. "Encoding time series as multi-scale signed recurrence plots for classification using fully convolutional networks." *Sensors* 20 (14): 3818.

Zhu, Hao. 2021. *kableExtra: Construct complex table with kable and pipe syntax*. R package version 1.3.4, `https://CRAN.R-project.org/package=kableExtra`.