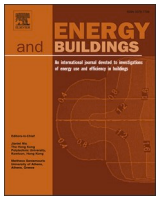
Energy & Buildings 355 (2026) 117043

Contents lists available at ScienceDirect

Energy & Buildings

journal homepage: www.elsevier.com/locate/enb



Good practices for documenting AI-based studies on energy and buildings

Tianzhen Hong *, Han Li

Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

A B S T R A C T

Artificial intelligence has transformed building science research over the past decade, with applications spanning energy modeling, energy prediction, HVAC optimization and controls, fault detection, and occupancy modeling. However, many studies lack adequate documentation of datasets, algorithms, training procedures, and validation methods. Building science research faces additional challenges including inconsistent evaluation metrics, limited generalizability across building types, climates, and significant gaps between experimental studies and deployed systems. This communication provides practical guidance for good practices in documenting and publishing AI-based research following established standards from the computer science and machine learning communities. By adopting frameworks such as Datasheets for Datasets, Model Cards, and standardized reproducibility checklists, researchers can ensure their work meets the rigorous documentation standards necessary for reproducible, comparable, and impactful building science research.

1. Introduction: The Growing use of AI in building science

Artificial Intelligence (AI) methods have achieved rapid adoption in building science research, with applications demonstrating 10–40 % improvements over traditional approaches [1]. Systematic reviews document successful machine learning (ML) applications across energy modeling and prediction [2,3], HVAC control, fault detection and diagnosis, occupancy modeling, and building performance optimization [4,5]. Deep learning approaches have proven particularly effective for time-series energy forecasting, with hybrid and ensemble methods providing the highest robustness [6]. For HVAC systems, which account for approximately 40 % of building energy consumption, AI-driven control strategies can reduce energy use by up to 40 % through dynamic adaptation [7].

Despite these promising results, significant challenges hinder the field from realizing ML's full potential. A critical issue is that most building science AI studies remain in the simulation environment, early prototype experimental or testing stage with limited real-world deployment and post-occupancy evaluation [8]. Unlike the computer science and machine learning communities, which have developed standardized documentation practices and reproducibility requirements, building science research lacks consistent reporting standards. Different studies use incompatible evaluation metrics, making performance comparison difficult [9]. Furthermore, all buildings are inherently different in design and operations, so models and control strategies developed for one building can be difficult to transfer to others [10].

The broader machine learning community has confronted these challenges through mandatory reproducibility requirements at major conferences. A landmark Princeton University study documented that data leakage and inadequate documentation affected 648 papers across 30 scientific fields [11]. In response, venues like NeurIPS (Conference on Neural Information Processing Systems), ICML (International Conference on Machine Learning), and ICLR (International Conference on Learning Representations) now require authors to complete comprehensive reproducibility checklists covering datasets, algorithms, training procedures, and computational resources [12]. When NeurIPS introduced its checklist in 2021, code sharing rates jumped from under 50 % to approximately 75 % [13].

This communication provides practical guidance for documenting AI-based building science research (Fig. 1). We synthesize established best practices from the computer science and machine learning communities and adapt them to the specific needs of energy and building research. Our goal is to help researchers produce studies that are reproducible, comparable, and build systematically on prior work. Note that this communication does not cover the use of AI tools in writing or refining manuscripts which is a totally different topic worth its own attention.

2. Critical gaps in current building science AI Publications

Building science AI studies consistently lack fundamental documentation needed to evaluate, reproduce, or build upon published work. The following gaps prevent meaningful comparison, performance

* Corresponding author.
E-mail address: thong@lbl.gov (T. Hong).

https://doi.org/10.1016/j.enbuild.2026.117043
Received 13 November 2025; Received in revised form 5 January 2026; Accepted 19 January 2026
Available online 20 January 2026

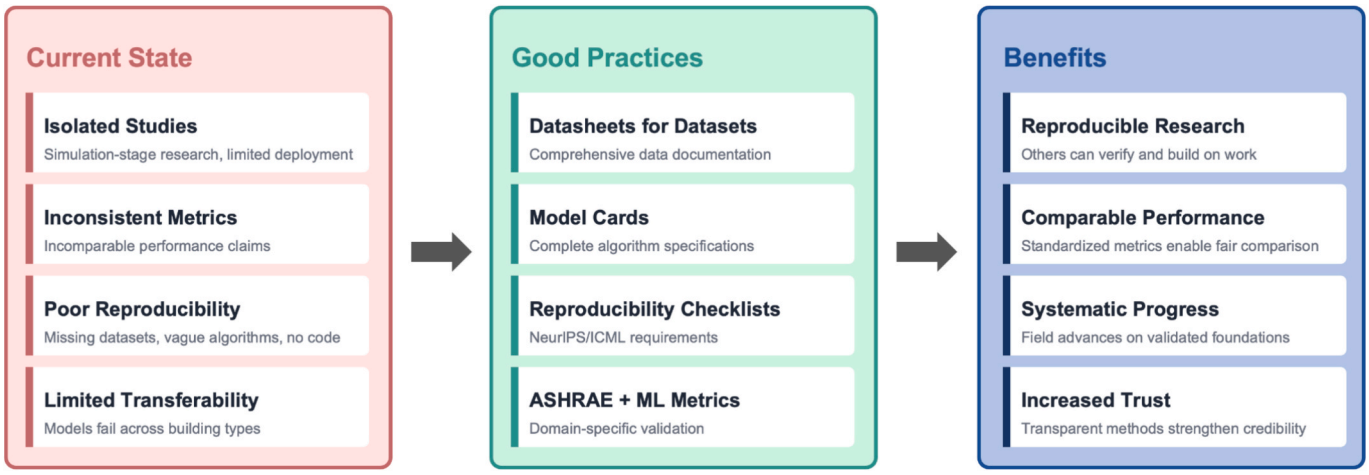


Fig. 1. Documentation framework bridging current building science AI practices with ML community standards.

verification, and identification of methods that actually work best for specific applications.

- **Dataset documentation:** Missing information about data sources, building types, climate zones, temporal resolution, preprocessing, and train/validation/test splits.
- **Algorithm and training details:** Vague descriptions without architecture, hyperparameters, optimization settings, or selection rationale.
- **Model evaluation:** Single-metric reporting without baselines, cross-validation, or error analysis.
- **Reproducibility:** Rare code/data availability and unreported random seeds that cause two-fold performance variation.

3. Good practices for AI-Based building science research

3.1. Dataset documentation

Comprehensive dataset documentation is foundational to reproducible AI research. The Datasheets for Datasets framework, now required by major ML conferences, provide a structured approach through seven sections: motivation, composition, collection process, preprocessing, uses, distribution, and maintenance [14]. For building science research, essential documentation includes data sources and collection methods (building management systems, IoT sensors, utility meters, or simulations), building characteristics (types, sizes, vintages, climate zones, HVAC configurations), and measurement specifications.

Dataset characteristics require quantitative descriptions: total data points, time span, temporal resolution, number and types of features, and target variable distributions. Building energy datasets often exhibit seasonal patterns, occupancy variations, and extreme values during anomalous conditions that should be characterized. Preprocessing steps must detail data cleaning procedures, outlier handling, missing data patterns and imputation approaches, normalization methods, and feature engineering [15].

Data splitting strategy is critical for valid evaluation. For time-series data, use temporal splitting where training precedes validation and test data chronologically. Document exact split ratios, dates, and how validation data was used. Data availability following the Findable, Accessible, Interoperable, and Reuseable (FAIR) principles is increasingly expected [16]. When proprietary data cannot be shared, provide synthetic datasets with similar properties, clear data request procedures, or detailed generation methods.

Data privacy and ethical considerations require explicit documentation when research involves human subjects or identifiable building data. Studies collecting occupant activities or behavior, conducting surveys, or using smart home data typically require Institutional Review Board (IRB) approval and informed consent. It is necessary to document IRB status, consent procedures, de-identification methods, and compliance with data protection regulations. When privacy concerns prevent data sharing, provide statistical summaries, synthetic datasets preserving key distributional properties, or secure data access protocols. Building-specific data should remove identifying characteristics unless explicit disclosure permission exists from owners.

3.2. Algorithm selection and documentation

Model Cards provide a comprehensive framework for documenting ML models [17]. Algorithm identification requires precise specification: for neural networks, document architecture including layers, units, activation functions, and parameter count; for ensemble methods, specify estimators, tree depth, and combination strategy; for support vector machines, document kernel type and parameters.

Hyperparameter documentation must be complete, listing all values and explaining whether they came from systematic tuning, literature recommendations, or defaults. Document the search space explored, search method used (grid search, random search, Bayesian optimization), and validation procedure [18]. Algorithm justification should explain why the selected approach is appropriate for the specific problem, discussing strengths and limitations in the context of building data characteristics.

Baseline comparisons are essential. Always include at least three baselines: a naive model (persistence for forecasting), a traditional statistical method (linear regression or ARIMA), and where possible, prior published methods [19]. Apply equal optimization effort to baselines. Document software implementation with specific library names and version numbers (e.g., scikit-learn 1.3.0, PyTorch 2.1.0).

3.3. Model training process

Training procedure documentation enables reproducibility. Document the optimization algorithm (e.g., Adam, SGD, RMSprop) with learning rate, momentum terms, learning rate schedule, batch size, and data shuffling. Regularization techniques critically affect performance: document dropout rates and locations, L1/L2 regularization coefficients, early stopping criteria including patience values and monitoring metrics, and any data augmentation methods.

Report training metrics for both training and validation sets. Learning curves showing loss versus epochs help diagnose underfitting, overfitting, or optimal capacity. Document convergence behavior including total epochs, whether early stopping triggered, and final performance gaps. Computational resources have become standard

Table 1
Documentation Requirements for Common Building Science AI Applications.

AI/ML Applications	Data Requirements	Model Documentation	Training Process	Evaluation & Results
Short-term Load Forecasting: Hourly electricity prediction for demand response (supervised regression)	3 offices, CZ 3A, 50–100 k ft^2 35 k/5.8 k/2.9 k temporal split (1 yr/2mo/1mo) Load, temp, humidity, time features Min-max norm, 24hr lags 3.2 % missing (linear interpolation < 6hr)	− LSTM: 2 layers (128,64) + dense (32,1) − 89 k params, ReLU/ linear activation − LR = 0.001, batch = 64, lookback = 168hr − Bayesian opt (50 trials) − Baselines: persistence, hist avg, linear − PyTorch 2.1.0, Python 3.10	− Adam (β1 = 0.9, β2 = 0.999) − Dropout 0.2, L2 λ = 0.001, early stop p = 15 − 150 epochs → stopped at 97 − V100 GPU, 2.3hrs − Training loss plateau at ep85	− CVRMSE = 12.3 % (ASHRAE < 30 % ✓) − RMSE = 45.2 kW, MAE = 32.1 kW − Walk-forward CV (7-day periods) − 67 % better vs persistence, 34 % vs linear − Errors high in extreme weather − Seeds: NumPy = 42, PyTorch = 42
HVAC Fault Detection: Classify AHU operating states for predictive maintenance (supervised classification)	− 5 AHUs, 2 institutional, VAV systems − 52 k normal / 17 k faults, 80/20 stratified − 9 classes (1 normal + 8 fault types) − Supply/return/outdoor temps, fan, damper, valve − 78 % normal, 22 % faults (imbalanced) − Z-score norm, SMOTE oversampling	− Random Forest: 500 trees, depth = 20 − Min split = 10, min leaf = 5 − 6 sensors + 12 engineered features − Grid search, 5-fold stratified CV − Baselines: logistic, decision tree, RP-1312 rules − SHAP for interpretability − scikit-learn 1.3.0, imbalanced-learn 0.11.0	− Tree-based, no gradient optimization − Deterministic given seed − 8.5 min training − 16-core CPU − Pruning via hyperparameters	− Accuracy = 94.2 %, F1-macro = 89.7 % − False alarm = 4.8 %, missed = 3.2 % − 5-fold stratified CV − 12 % better F1 vs logistic, 8 % vs rules − Confusion between similar faults − Tested 2 new buildings: F1 = 81.3 % (13 % drop) − Seed: scikit-learn = 123
Occupant Thermostat Behavior Clustering: Identify user archetypes for personalized control (unsupervised clustering)	− 156 apartments, smart thermostats, 6 months − 89 k setpoint events (changes, overrides) − Features: change freq, magnitude, timing, outdoor temp correlation, day patterns − No labels (unsupervised) − Z-score norm per feature − Full dataset (no train/test split)	− k-means clustering − k = 4 (elbow method, silhouette analysis) − Euclidean distance, k-means++ init − Features: 12 behavioral metrics − Baselines: k = 3,5,6 for comparison − PCA for visualization (2D projection) − scikit-learn 1.3.0, Python 3.10	− Lloyd's algorithm, 300 max iterations − Converged at 47 iterations − Multiple random inits (n = 50), best kept − 2 min training − CPU only − Deterministic clustering achieved	− Silhouette score = 0.61 (good separation) − Davies-Bouldin index = 0.74 (lower better) − Elbow at k = 4 (variance explained) − 4 archetypes: set-and-forget (41 %), frequent adjusters (28 %), temp-sensitive (19 %), schedulers (12 %) − Validated via user surveys (82 % match) − Seeds: NumPy = 42, k-means inits = 50
HVAC Optimal Control: Learn chiller/ pump policy to minimize cost + maintain comfort (reinforcement learning)	− 100 k ft^2 office, CZ4A, chilled water system − 2 yr building data + EnergyPlus 23.2.0 − Calibrated sim: CVRMSE < 15 % − State: 24-dim (temps, weather, time, equip) − Action: continuous (setpoint 5–12 °C, pump 30–100 %) − Reward: −(cost + 100 × discomfort) − 1000 episodes × 96 steps (24hr each)	− TD3 (Twin Delayed DDPG) − Actor: 3 × FC (256,256, action), tanh output − Critics: 2 × [3 × FC (256,256,1)] − Actor 131 k params, each Critic 132 k − γ = 0.99, τ = 0.005, delay = 2, noise σ = 0.1, buffer = 1 M − Baselines: rule-based BMS, MPC − PyTorch 2.1.0, Gym 0.26.2, EPPY 0.5.63	− Adam for actor/critics (LR = 3e-4) − Gaussian noise σ = 0.1 → 0.01 annealed − 2 M env steps, 180hrs − A100 GPU − Random episode starts − Target networks for stability	− 23 % cost reduction vs rule-based, 8 % vs MPC − 98.7 % hours in comfort (21–24 °C) − 30-day test (unseen weather) − Robustness: 12 % degradation at ± 1°C noise, 18 % at forecast errors − Inference: 0.03 s/step (real-time ok) − Mean ± std over 5 seeds − Seeds: env = 2024, agent = 42
Load Shape Generation: Synthesize realistic building load profiles for design/ planning (generative modeling)	− 2400 daily load profiles (100 buildings × 24 days) − Commercial offices, CZ4A, 20–150 k ft^2 − 24-hour profiles at 1hr resolution − Features: 24 hourly loads (kW) − 80/20 train/val split by building − Min-max norm to [1] per profile	− DCGAN (Deep Convolutional GAN) − Generator: latent z(100) → 4 up-conv layers → 24 values − Discriminator: 24 inputs → 4 conv layers → real/ fake − G: 89 k params, D: 76 k params − LeakyReLU (α = 0.2), BatchNorm, tanh output − Baselines: Gaussian mixture, VAE − PyTorch 2.1.0, Python 3.11	− Adam for G/D (LR = 2e-4, β1 = 0.5, β2 = 0.999) − Alternating: 1 G update per 5 D updates − 10 k iterations − Batch = 64, label smoothing (0.9/0.1) − RTX 3090, 3.5hrs − Mode collapse monitored via diversity	− Fréchet Inception Distance (FID) = 23.4 − Inception Score (IS) = 2.8 ± 0.2 − Visual inspection: realistic shapes − Statistical tests: KS test p > 0.05 (dist match) − Diversity: 87 % unique peak hours − Domain expert validation: 92 % realistic − Failed: extreme weather days underrepresented − Seeds: PyTorch = 2024, latent sampling = 42
Building Code Q&A Assistant: Answer technical questions on building standards for compliance checking (Supervised LLM fine-tuning)	− 12 k Q&A pairs from standard − 8.4 k/2.1 k/1.5 k random split (70/17.5/12.5) − Questions: avg 24 tokens,	− Base: Llama-3.1-8B (8B params) − PyTorch 2.1, transformers 4.36 − LoRA fine-tuning: r =	− AdamW optimizer (LR = 2e-4, β1 = 0.9, β2 = 0.999) − Cosine LR schedule, warmup 100 steps	− Exact match = 34.2 %, ROUGE-L = 71.8 % − BERTScore F1 = 82.4 % − Human expert eval: 78 % correct, 89 % helpful

(*continued on next page*)

Table 1 (*continued*)

AI/ML Applications	Data Requirements	Model Documentation	Training Process	Evaluation & Results
	max 128 – Answers: avg 156 tokens, cite sections – Context: RAG-retrieved standard sections (top-3, max 2048 tokens) – Curated from code interpretations, FAQs, expert answers	16, α = 32, dropout = 0.05 – Context length: 4096 tokens – Baselines: GPT-4 zero-shot, base Llama-3.1, retrieval-only	– Gradient accumulation: 4 steps, effective batch = 32 – 3 epochs (787 steps), BF16 precision – Converged at epoch 2.4 – 4 × A100 GPUs (80 GB), 8.5hrs – Gradient clipping: max norm = 1.0	– 67 % better than base, 12 % below GPT-5 – 5-fold CV on question types (design/prescriptive/performance) – Failures: multi-standard questions, calculations – Inference: 2.3 s/query (A100)

reporting requirements: document hardware specifications, total training time, and estimated energy consumption where feasible.

3.4. Testing and validation

Building science requires multiple evaluation metrics since no single metric captures all performance aspects or application context. For energy prediction, ASHRAE Guideline 14 establishes CVRMSE and MBE as standard metrics [20]. For monthly data, acceptable calibration requires CVRMSE within 15 % and MBE within 5 %; for hourly data, thresholds are 30 % and 10 %. Complementary metrics to report include RMSE, MAE, MAPE, and R^2.

For classification tasks, accuracy alone is insufficient for imbalanced datasets. Reports should include precision, recall, F1 score, and confusion matrices. Practical implications of false positives versus false negatives should be discussed. Cross-validation provides robust performance estimates. Use k-fold cross-validation (e.g., 5 or 10 folds). For time-series data, use temporal approaches that maintain ordering.

Error analysis should examine model behavior across different conditions: seasons, weather extremes, occupancy levels, and equipment modes. Identify where models perform poorly and discuss practical limitations. Generalization assessment is critical since building models often fail to transfer. Evaluate on unseen buildings when possible and document characteristics where models succeed versus fail.

3.5. Reproducibility requirements

Code availability has become standard practice, with major conferences achieving approximately 75 % code sharing rate [13]. Share complete repositories including preprocessing scripts, sample data with expected outputs for environment verification, training code, evaluation scripts, and model weights. Provide clear README files with installation instructions and dependencies. Deposit code on permanent archival repositories with DOI (e.g., Zenodo, Dryad, Dataverse) for long-term accessibility [21].

Random seed documentation is critical since changing seeds can inflate performance by up to two-fold [22]. Set and report seeds for data splitting, initialization, and training. Specify random number generator libraries used and whether multiple runs characterized variability. Software dependencies must include version numbers since different versions produce different results. Create requirements files specifying exact versions and document computational environment including operating system and GPU framework version (e.g., CUDA, ROCm, OneAPI). For neural networks, provide complete architecture definitions or trained weights when feasible and training is computationally expensive.

3.6. Additional considerations for foundation models

Foundation models and large language models require additional documentation when used in building science research such as analyzing maintenance logs, processing building documentation, or natural language interfaces. Document the specific model version (e.g., GPT-5, Claude 4.5, DeepSeek v3.2), API endpoint or local deployment, prompt engineering approach including system prompts and few-shot examples, and sampling parameters (e.g., temperature, top-p). Since many foundation models are accessed via APIs rather than trained from scratch, focus documentation on fine-tuning procedures if applicable, retrieval-augmented generation (RAG) architectures if used, and evaluation on domain-specific tasks. Note that even with fixed random seeds, API-based models may produce non-deterministic outputs due to infrastructure changes.

4. Examples of documentation for common building science ML applications

Table 1 provides concrete documentation examples across five common building science ML applications, demonstrating the level of details and specificity required for reproducible research. These examples span supervised learning, unsupervised learning, reinforcement learning, and generative modeling paradigms.

5. Summary

Standardized documentation practices would significantly strengthen building science research using machine learning methods. Adopting established frameworks from the ML community helps transform isolated experiments into systematically comparable studies that advance the field. The documentation framework outlined in this article (comprehensive dataset descriptions, complete algorithm specifications, detailed training procedures, and rigorous evaluation protocols) enables other researchers to verify, reproduce, and build upon published work. As AI methods continue evolving toward more sophisticated architectures and larger-scale applications, documentation standards will need to evolve accordingly.

CRediT authorship contribution statement

Tianzhen Hong: Writing – review & editing, Writing – original draft, Supervision, Methodology, Conceptualization. **Han Li:** Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The corresponding author Tianzhen Hong is the Executive Editor of Energy and Buildings. This manuscript will be independently handled by other editors and Dr. Hong would not be able to access the review process of this manuscript to ensure a fair review.

Acknowledgments

This work was supported by the United States Department of Energy under Contract No. DE-AC02-05CH11231. Any opinions, findings, and

conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Energy.

Data availability

No data was used for the research described in the article.

References

[1] S. Seyedzadeh, F.P. Rahimian, I. Glesk, M. Roper, Machine learning for estimation of building energy consumption and performance: a review, Vis Eng 6 (2018) 5, https://doi.org/10.1186/s40327-018-0064-7.

[2] T. Hong, L. Zhang, AI for building energy modeling: a transformation, Build. Simul. 18 (2025) 2219–2225, https://doi.org/10.1007/s12273-025-1329-4.

[3] H. Li, Y. Xu, T. Hong, EnergyPlus-MCP: a model-context-protocol server for AI-driven building energy modeling, SoftwareX 32 (2025) 102367, https://doi.org/10.1016/j.softx.2025.102367.

[4] M. Huotari, A. Malhi, K. Främling, Machine Learning applications for Smart Building Energy utilization: a Survey, Arch. Comput. Methods Eng. 31 (2024) 2537–2556, https://doi.org/10.1007/s11831-023-10054-7.

[5] T. Hong, Z. Wang, X. Luo, W. Zhang, State-of-the-art on research and applications of machine learning in the building life cycle, Energy Build. 212 (2020) 109831, https://doi.org/10.1016/j.enbuild.2020.109831.

[6] S. Ardabili, L. Abdolalizadeh, C. Mako, B. Torok, A. Mosavi, Systematic Review of Deep Learning and Machine Learning for Building Energy, Front. Energy Res. 10 (2022), https://doi.org/10.3389/fenrg.2022.786027.

[7] S.L. Zhou, A.A. Shah, P.K. Leung, X. Zhu, Q. Liao, A comprehensive review of the applications of machine learning for HVAC, DeCarbon 2 (2023) 100023, https://doi.org/10.1016/j.decarb.2023.100023.

[8] P.W. Tien, S. Wei, J. Darkwa, C. Wood, J.K. Calautit, Machine Learning and Deep Learning Methods for Enhancing Building Energy Efficiency and Indoor Environmental Quality – a Review, Energy AI 10 (2022) 100198, https://doi.org/10.1016/j.egyai.2022.100198.

[9] Z. Chen, F. Xiao, F. Guo, J. Yan, Interpretable machine learning for building energy management: a state-of-the-art review, Adv. Appl. Energy 9 (2023) 100123, https://doi.org/10.1016/j.adapen.2023.100123.

[10] H.P. Das, Y.-W. Lin, U. Agwan, L. Spangher, A. Devonport, Y. Yang, et al., Machine Learning for Smart and Energy-Efficient buildings, Environ. Data Sci. 3 (2024) e1.

[11] S. Kapoor, A. Narayanan, Leakage and the reproducibility crisis in machine-learning-based science, Patterns 4 (2023) 100804, https://doi.org/10.1016/j.patter.2023.100804.

[12] NeurIPS. NeurIPS Paper Checklist Guidelines 2025. https://neurips.cc/public/guides/PaperChecklist.

[13] Pineau J, Vincent-Lamarre P, Sinha K, Larivière V, Beygelzimer A, d'Alché-Buc F, et al. Improving Reproducibility in Machine Learning Research (A Report from the NeurIPS 2019 Reproducibility Program) 2020. https://doi.org/10.48550/arXiv.2003.12206.

[14] Gebru T, Morgenstern J, Vecchione B, Vaughan JW, Wallach H, III HD, et al. Datasheets for Datasets 2021. https://doi.org/10.48550/arXiv.1803.09010.

[15] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, O. Tabona, A survey on missing data in machine learning, J Big Data 8 (2021) 140, https://doi.org/10.1186/s40537-021-00516-9.

[16] M.D. Wilkinson, M. Dumontier, A. IjJ, G. Appleton, M. Axton, A. Baak, et al., The FAIR Guiding Principles for scientific data management and stewardship, Sci. Data 3 (2016) 160018, https://doi.org/10.1038/sdata.2016.18.

[17] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, et al., Model Cards for Model Reporting, Proc. Conf. Fairness Account. Transpar. (2019) 220–229, https://doi.org/10.1145/3287560.3287596.

[18] J. Pineau, The Machine Learning Reproducibility Checklist (2020).

[19] Li D, Hasanaj E, Li S. 3 - Baselines. 3 – Baselines 2020. https://blog.ml.cmu.edu/2020/08/31/3-baselines/ (accessed November 10, 2025).

[20] American Society of Heating, Refrigerating and Air-Conditioning Engineers. ASHRAE Guideline 14-2014: Measurement of Energy, Demand, and Water Savings. Atlanta, GA: ASHRAE; 2014.

[21] Tips for Publishing Research Code 2021. https://github.com/paperswithcode/releasing-research-code (accessed November 10, 2025).

[22] A.L. Beam, A.K. Manrai, M. Ghassemi, Challenges to the Reproducibility of Machine Learning Models in Health Care, JAMA 323 (2020) 305–306, https://doi.org/10.1001/jama.2019.20866.