

# Methods of utilizing survey data for predicting building energy consumption using transformers and MLP with dataset merging, transformation, and DistilBERT

Qiyuan Sun<sup>\*1</sup>, Martin Soldani<sup>†2</sup>, Max Geddes<sup>2</sup>, and Zac Enviah<sup>2</sup>

<sup>1</sup>School of Civil Engineering, University of Sydney, Australia

<sup>2</sup>School of Computer Science, University of Sydney, Australia

July 27, 2025

## Abstract

This paper investigates optimal methods for utilizing survey data in building energy consumption prediction through AI models. While survey data presents unique challenges due to its unstructured and often inconsistent nature, this study focuses on refining preprocessing techniques and integrating standardized AI models to evaluate its potential effectively. We assess the performance of Transformer, Multi-Layer Perceptron (MLP), XGBoost, and Support Vector Machine (SVM) models using R-squared and Mean Absolute Error (MAE) metrics on RECS and CBECS datasets. Transformer models consistently outperformed the others due to their superior attention mechanisms, excelling in high-dimensional data scenarios. MLPs followed closely, while XGBoost and SVM struggled with the complexities of unprocessed survey data. Our research also highlights the impact of feature transformation, where DistilBERT embeddings significantly improved Transformer performance, while one-hot encoding posed difficulties for traditional models in managing high-dimensional data. Merging datasets based on common features further enhanced the performance of Transformers and MLPs by increasing dataset size and improving model learning. However, concerns of overfitting in Transformers and underfitting in MLPs indicate the need for regularization and architectural adjustments. This research demonstrates the potential of survey data as a viable resource for building energy prediction, especially when paired with appropriate preprocessing techniques and robust AI models, with Transformers and MLPs showing the most promise for practical deployment. Future research should continue to refine these techniques, particularly focusing on integrating specifically survey data into broader building energy prediction frameworks to enhance accuracy and applicability across various building types.

---

<sup>\*</sup>qsun0008@uni.sydney.edu.au

<sup>†</sup>martin.soldani@sydney.edu.au

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Key Terminologies . . . . .	6
<b>2</b>	<b>Related Work</b>	<b>7</b>
2.1	Machine learning studies with same dataset . . . . .	9
<b>3</b>	<b>Methodology</b>	<b>10</b>
3.1	Datasets . . . . .	10
3.2	Research Methods . . . . .	16
3.3	Data preprocessing . . . . .	19
3.3.1	Data transformation . . . . .	19
3.3.2	One hot . . . . .	20
3.3.3	Distil Bert vectorizer . . . . .	20
3.4	Data merging . . . . .	21
3.4.1	Merging by common columns . . . . .	22
3.4.2	Merging in full . . . . .	22
3.5	Inputs and Integrity . . . . .	23
3.5.1	Input validation . . . . .	23
3.5.2	Data leakage . . . . .	24
3.6	Models . . . . .	26
3.6.1	Model Architectures . . . . .	26
3.6.2	Transformer . . . . .	27
3.6.3	MLP . . . . .	28
3.6.4	Output and compile . . . . .	29
<b>4</b>	<b>Results</b>	<b>30</b>
4.1	R-squared Testing Results . . . . .	30
4.1.1	Models comparison . . . . .	30
4.1.2	Merging comparison . . . . .	33
4.1.3	Transformation comparison . . . . .	34
4.2	MAE Testing Results . . . . .	36
4.3	Transformer and MLP Comparison During Training . . . . .	38
4.4	Transformation, and Merging Methods During Training . . . . .	42
4.5	Model training results . . . . .	45
4.6	Model training time and inference time . . . . .	46
<b>5</b>	<b>Discussion</b>	<b>47</b>
5.1	Limitations and recommendations . . . . .	47
5.2	Future works . . . . .	49
<b>6</b>	<b>Conclusion</b>	<b>52</b>
<b>7</b>	<b>Availability</b>	<b>53</b>

<b>8 Acknowledgement</b>	<b>53</b>
<b>9 References</b>	<b>53</b>
<b>10 Appendix</b>	<b>57</b>
10.1 Training results . . . . .	57
10.2 Columns Dropped . . . . .	73
10.3 RECS target columns . . . . .	73

# 1 Introduction

Ever increasing global energy demands and a worrying trend of global warming have prompted many researchers worldwide to find new energy solutions while optimizing the energy efficiency of current implementations. This includes the building construction and maintenance sector, responsible for approximately 37% of global emissions [1]. As a result, there is a continuous push towards designing more sustainable buildings, with engineers and architects increasingly prioritizing energy-efficient building designs.

Accurately predicting building energy consumption is crucial in the design phase of a building, as such estimates are heavily influential in making decisions concerning the building’s energy efficiency. Early detection of energy inefficiencies enables the needed refinements to be made at a time when they can be less costly, and will prevent the need for more expensive modifications at later periods of the building construction or operation. But also throughout the other stages of a building’s life energy-use forecasting can deliver information on the probable energy cost, and thus enable resources to be managed more effectively.

Traditional energy simulation software models, while commonly used, often rely on numerous assumptions that can diminish the performance and validity of their calculations. In contrast, AI offers a data-driven approach that can provide more accurate and reliable estimates for end-users [2]. Various data sources have been employed in the literature for such building energy consumption predictions, with the most prominent being sensor-based data and simulation data. Sensor-based data involves real-time measurements from devices installed within buildings, capturing detailed information about energy usage, environmental conditions, and operational patterns. However, the collection of sensor data is often expensive and requires extensive infrastructure, limiting its scalability for larger datasets or widespread applications. Simulation data, on the other hand, is generated through advanced software tools such as EnergyPlus [3]. These tools create theoretical models of building performance, simulating energy consumption under various operational scenarios. While these simulations offer a flexible and detailed approach to predicting energy use, they rely heavily on assumptions about building occupancy, climate conditions, and operational behaviors. Consequently, their predictive power is limited by the accuracy of these assumptions, potentially leading to discrepancies when applied to real-world settings. In contrast to these sources, survey data offers a potentially rich, yet often overlooked, source of information for energy predictions. Survey data typically includes detailed responses about building characteristics, appliance use, and occupant behaviors, which are important factors in determining energy consumption. This type of data is often collected through national surveys such as the Residential Energy Consumption Survey (RECS) and the Commercial Buildings Energy Consumption Survey (CBECS), offering broad coverage across regions and building types.

In this paper, we specifically investigate the potential of incorporating survey data into AI model training for building energy consumption predictions. Although numerous studies in the literature have demonstrated the efficacy of artificial intelligence in this domain, survey data is often overlooked due to the unique challenges it presents. The lack of standardized constraints in survey responses can lead to inconsistencies or incomplete entries, complicat-

ing meaningful data extraction by machine learning (ML) models and necessitating extensive data cleaning efforts. However, we seek to demonstrate that despite these challenges, survey data can provide significant advantages and should not be dismissed. Survey data offers a highly accessible and easily collectible source of information, especially when it pertains to objective building characteristics like size, layout, and energy systems. This objectivity reduces the potential for bias, offering a consistent and reliable dataset for training AI models. In other words it is more accessible and cost-effective compared to sensor data, which requires significant investment in equipment and infrastructure. Government organizations are particularly well-suited to conduct surveys, making it a reliable source of information. Surveys are an efficient way to collect large-scale data, offering insights into various aspects of building energy consumption. This approach allows for a more widespread and affordable means of gathering data across diverse regions and building types. Our research aims to highlight how leveraging survey data can enhance AI-based building energy consumption predictions and contribute valuable insights to the field. Such that survey designers create more efficient and valuable surveys, while enabling researchers to leverage this data to enhance the AI-driven building energy prediction sector, which often suffers from a lack of comprehensive datasets.

A key component of this paper also explores the effects of data augmentation techniques, particularly the merging of related dataset rows based on common features, and its impact on final model performance. Merging increases data inclusivity and training set size, which, in turn, can enhance model robustness. Furthermore, to fully harness the potential of survey data and contribute to the broader research efforts around data cleaning, we delve into the use of vectorizers to standardize survey responses. By applying advanced natural language processing (NLP) techniques, we aim to address inconsistencies in survey responses, ensuring a more uniform dataset that mitigates issues related to non-standardized entries.

By investigating the best techniques to process and enhance survey data, we aim to set a precedent for making survey data a more effective and viable tool for building energy prediction in future research. Our objective is to determine the most effective methods to process survey data for machine learning purposes, identifying strategies that improve prediction accuracy and model robustness. Through systematic comparisons of different AI models, such as Transformers, Multi-Layer Perceptrons (MLP), Extreme Gradient Boosting (XGBoost), and Support Vector Machines (SVM), trained on preprocessed survey datasets, we aim to demonstrate how survey data can be transformed into a valuable asset for building energy consumption predictions.

## 1.1 Key Terminologies

Abbreviation	Full Name
MLP	Multilayer Perceptron
DistilBERT	Distilled Bidirectional Encoder Representations from Transformers
MSE	Mean Squared Error
MAE	Mean Absolute Error
ML	Machine Learning
DL	Deep Learning
AI	Artificial Intelligence
ANN	Artificial Neural Network
DOE	Department of Energy
CBECS	Commercial Building Energy Consumption Survey
RECS	Residential Energy Consumption Survey
DAST	Dynamic Application Security Testing
GAN	Generative Adversarial Network
LinearR	Linear Regression
RFreg	Random Forest Regressor
Bagging	Bootstrap Aggregating
MLPreg	Multilayer Perceptron Regressor
XGB	Extreme Gradient Boosting
XGBcommon	Extreme Gradient Boosting for Common Tasks
EUI	Energy Use Intensity
LassoR	Lasso Regression
SVM	Support Vector Machine
GB	Gradient Boosting
GNB	Gaussian Naive Bayes
RF-Classifer	Random Forest Classifier
KNN	K-Nearest Neighbors
LogisticR	Logistic Regression
LS	Least Squares
StepwiseAIC	Stepwise Akaike Information Criterion
EIA	Energy Information Administration
PLS	Partial Least Squares
ElasticNet	Elastic Net
GGB	Generalized Gradient Boosting
XGBoost	Extreme Gradient Boosting
RF	Random Forest
NLP	Natural Language Processing
FFN	Feedforward Neural Network
NN	Neural Network

## 2 Related Work

The field of predicting energy consumption is growing steadily, with researchers placing their biggest focus on finding appropriate AI models and improving upon them. Y. Wei [4] concluded that ANN learning models are currently outperforming other models in most types of building-related energy predictions. Wang and Srinivasan [5] analysed the advantages and limitations of different AI-based prediction models, including ensemble methods, concluding that model performance highly depends on the use case. While ensemble methods promised the highest performance, they suffer from slow training time and require a high level of expertise. Each model involves trade-offs, and hence one must always optimize for the use case. Our paper, however, is more concerned with survey datasets — specifically, to what extent and how they can be used to maximize the potential learning process. According to Amasyali and El-Gohary [2], who reviewed the field of building energy consumption prediction for data-driven AI approaches, too little focus is placed on the impact of dataset selection, including the potential of survey datasets.

Recent studies have begun to highlight the importance of dataset selection in building energy consumption prediction. Chen, Singh, and Geyer [6] introduced Component-Based Machine Learning (CBML), a method that integrates domain knowledge to enhance machine learning robustness with small, inconsistent datasets. Their approach reduces data dependency while improving model interpretability and performance. Pipattanasomporn *et al.* [7] provided the CU-BEMS dataset, which includes detailed electricity consumption and environmental data from a seven-story office building. The high-resolution data supports a range of applications, such as load forecasting and thermal modeling. They have demonstrated the value of granular datasets in improving prediction accuracy. Mathew *et al.* [8] compiled the DOE Buildings Performance Database, discussing challenges in consolidating data from over 750,000 buildings. They emphasized the importance of uniformity and quality in large-scale datasets to ensure reliability in energy performance analysis and peer group comparisons. Zhang *et al.* [9] introduced a similarity-based grouping method to enhance short-term cooling load predictions by clustering data based on weather and time similarities. Their method improved model accuracy even in the absence of occupancy data, showing how dataset structure can significantly impact prediction outcomes. Xiao *et al.* [10] organized the "Energy Detective" competition to evaluate building energy prediction methods with limited data. The competition highlighted the potential of hybrid models, combining multiple models for greater accuracy and interpretability. In data-driven approaches, the authors stressed the importance of improving data preparation, including anomaly handling and feature engineering. They emphasized the key role of dataset selection in cross-building energy predictions and recommended refining data preprocessing for better model performance, especially for error-prone data such as survey data.

The impact of datasets extends far beyond any single application, with recent research underscoring the critical role dataset quality plays in machine learning performance. This includes the quality of survey-based datasets, as highlighted by Gong *et al.* [11]. Their study outlines a framework for evaluating dataset quality, emphasizing dimensions such as accuracy, completeness, and consistency. This structured approach ensures that survey data, like

other datasets, can support reliable model training and improve overall predictive performance. Lee *et al.* [12] emphasized the importance of data cleaning, particularly for survey datasets prone to errors and inconsistencies. They discussed methods that remove noise from data, enhancing the accuracy and reliability of models trained on survey-based information. Paullada *et al.* [13] explored the impact of poor dataset practices, such as bias in survey data, which can lead to skewed predictions and negative outcomes. They advocate for better documentation and filtering processes to mitigate bias, ensuring survey datasets contribute positively to machine learning applications.

Dataset augmentation is a critical technique to address data scarcity, especially in the context of building energy consumption prediction. Mumuni *et al.* [14] provided an extensive review of augmentation techniques primarily used in computer vision, such as generative adversarial networks (GANs) and feature-level augmentation. However, their application to survey-based datasets remains largely unexplored. In building energy consumption prediction, recent work by Fang *et al.* [15] introduced the Data Augmentation based on Occupancy Behavior (DAOB) method, which improved model performance by simulating future occupancy scenarios beyond historical limits. Their case study demonstrated a significant reduction in prediction errors, particularly in challenging scenarios, highlighting the potential of combining physical and data-driven models. Deng *et al.* [16] further advanced data augmentation by proposing a decomposition-based scheme (DAST) for time-series building load data. This approach decomposes time-series data into components, enabling the generation of more accurate synthetic data in cases with limited time coverage. Their method outperformed traditional GAN-based schemes, reducing errors by 47.8% and significantly improving the performance of building load forecasting tasks. While these techniques show promise in augmenting building energy data, the application of dataset merging and augmentation in survey-based datasets for building energy consumption remains novel.

In summary, while significant advancements have been made in improving AI models for building energy consumption prediction, the optimization of methods specifically designed to handle survey-based data remains under-explored. Recent studies have underscored the importance of data quality, augmentation, and preparation, yet there is a notable gap in the literature regarding the integration of advanced preprocessing techniques with standardized AI model comparison on modified survey datasets. Our research brings valuable insights by refining survey data handling and applying a consistent set of AI models to rigorously evaluate the effectiveness of these methods, allowing us to assess how survey data transformation and augmentation impact model performance. With this paper, we aim to bring more attention to the potential of survey data in enhancing model performance for data-driven building energy predictions.



## 2.1 Machine learning studies with same dataset

Study	Model Used	Results
<b>CBECS</b>		
Machine learning approaches for estimating commercial building energy consumption (2017) [17]	Models trained on CBECS	$R^2$ validated on New York City Local Law 84 dataset:
	LinearR	$0.88 \pm 0.01$
	RFreg	$0.87 \pm 0.01$
	Bagging	$0.87 \pm 0.02$
	MLPreg	$0.64 \pm 0.03$
	XGB	$0.89 \pm 0.01$
	XGBcommon	$0.82 \pm 0.02$
Predictive modeling for US commercial building energy use: A comparison of existing statistical and machine learning algorithms using CBECS microdata (2018) [18]	EUI prediction	MAE and variance:
	LinearR	$29.7 \pm 1.6$
	LassoR	$27.1 \pm 1.3$
	SVM	$25.7 \pm 1.1$
	ANN	$27.1 \pm 0.9$
	GB	$26.7 \pm 1.0$
	RF	$27.0 \pm 1.1$
Predicting Energy Consumption in Commercial Buildings using its Property features and Machine Learning Algorithms (2019) [19]	PCA:	Accuracy:
	GNB	93.37%
	RF-Classifer	96.50%
	KNN	98.15%
	LogisticR	70.97%
<b>RECS</b>		
How do machines predict energy use? Comparing machine learning approaches for modeling household energy demand in the United States (2022) [20]	Models:	$R^2$ :
	LS	0.6142
	StepwiseAIC	0.6074
	PLS	0.6180
	KNN	0.5173
	ElasticNet	0.6010
	GGB	0.6313
	XGBoost	0.6554
	RF	0.6426

Table 1: Summary of Related Studies, Models Used, and Results

This table presents machine learning studies that utilize the same dataset as ours, pro-

viding a basis for benchmarking our results. Benchmarking against existing literature helps validate our models by demonstrating comparable or improved performance, thus confirming the robustness and relevance of our approach.

### 3 Methodology

This paper aims to analyze methods for utilizing survey data to develop robust building energy prediction models. It builds on existing literature by comparing results from studies that use survey data for building energy prediction with our findings. This comparison validates our model architectures and implementations as meeting or exceeding current literature standards. From there, we explore two key hypotheses: first, merging datasets based on common rows does not result in significant loss of information, and second, it is possible to use an NLP encoder to process uncommon rows so that all data is effectively utilized. Merging datasets can be beneficial as it increases the training data size and expands data inclusivity, leading to more robust models.

To achieve this, we begin by cleaning datasets (visualize cleaning by producing visual aids before and after) such as CBECS and RECS, training base models on each. We then merge and augment the datasets, train models on the merged data, and apply a vectorizer to the text features to assess performance improvements. The results will allow us to draw conclusions about the best methods and provide recommendations. To demonstrate the practical value of our research, we additionally plan to develop a website for survey-based energy prediction, see the Availability section.

#### 3.1 Datasets

The data is collected from U.S. Energy Information Administration (EIA) and includes the following: 2020 RECS survey microdata (RECS20) [21], 2015 RECS Survey microdata (RECS15) [22], 2018 CBECS Survey microdata (CBECS18) and code book [23], and 2012 CBECS Survey microdata (CBECS12) and code book [24].

The Residential Energy Consumption Survey (RECS) and the Commercial Buildings Energy Consumption Survey (CBECS) are large-scale national surveys conducted by the U.S. Energy Information Administration (EIA) to collect data on energy usage in residential and commercial buildings, respectively. RECS focuses on energy consumption patterns, housing characteristics, and energy expenditures in residential homes across the United States. The data includes variables related to household characteristics, such as income, housing type, heating and cooling systems, energy sources, and appliances. This dataset provides insights into how different factors, like geographic location or home size, impact energy usage in homes, making it a valuable resource for understanding energy trends in the residential sector.

CBECS, on the other hand, gathers data on energy consumption and building characteristics in the commercial sector, which includes offices, retail spaces, hospitals, schools, and other non-residential buildings. This survey captures information on building size, energy

systems, energy sources, operational characteristics, and total energy usage. CBECS helps to analyze how various building types and operational behaviors contribute to energy consumption, supporting energy policy development and building energy efficiency standards. Both datasets are critical for researchers, policymakers, and businesses interested in optimizing energy use and understanding consumption trends in the U.S. building sector.

The following shows a table that generally describes and a visualization of the normalized values for each dataset

<b>Dataset: recs15</b>	<b>Statistics</b>	
Total Entries	5,686	
Total Columns	704	
Columns of type float64	208 (Continuous numeric data)	
Columns of type int64	492 (Discrete numeric data)	
Columns of type object	4 (Likely categorical or string)	
Missing Values	2,382	
Memory Usage	31.72 MB	
<b>Unique Values per Object Column</b>		[H]
METROMICRO	3 unique values	
UATYP10	3 unique values	
CLIMATE_REGION_PUB	5 unique values	
IECC_CLIMATE_PUB	11 unique values	
<b>Target Columns Statistics</b>		
KWH Mean	11,028.93	
KWH Standard Deviation	7,049.73	
KWH Minimum	59.078	
KWH Maximum	63,216.806	

Table 2: Statistics of Dataset recs15

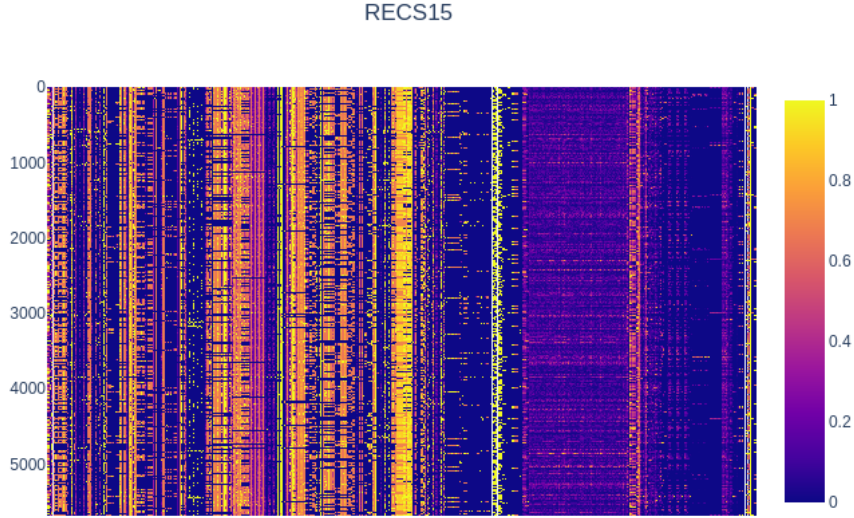


Figure 1: RECS15 normalized data visualization

Dataset: recs20	Statistics
Total Entries	18,496
Total Columns	743
Columns of type float64	188 (Continuous numeric data)
Columns of type int64	548 (Discrete numeric data)
Columns of type object	7 (Likely categorical or string)
Missing Values	653
Memory Usage	111.63 MB
<b>Unique Values per Object Column</b>	
REGIONC	4 unique values
DIVISION	10 unique values
state_postal	51 unique values
state_name	51 unique values
BA_climate	8 unique values
IECC_climate_code	17 unique values
UATYP10	3 unique values
<b>Target Columns Statistics</b>	
KWH Mean	10,848.82
KWH Standard Deviation	7,111.77
KWH Minimum	42.01
KWH Maximum	184,101.84

Table 3: Statistics of Dataset recs20

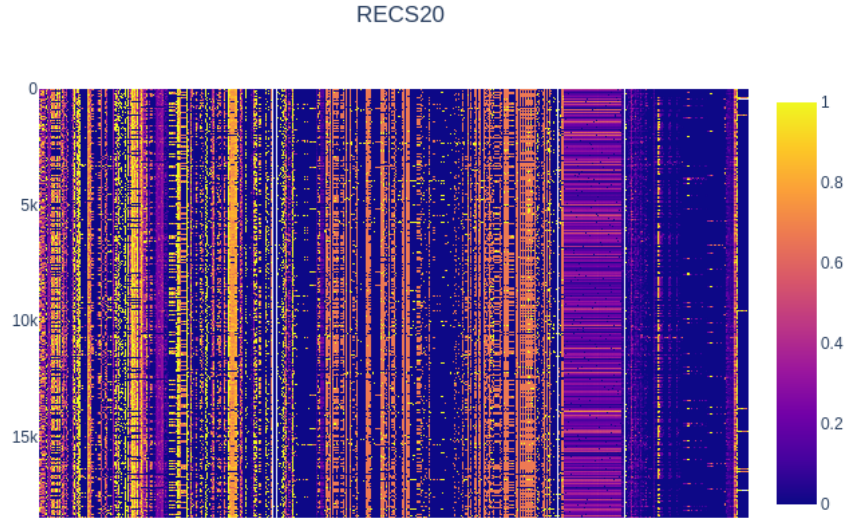


Figure 2: RECS20 normalized data visualization

Dataset: cbees12	Statistics
Total Entries	6,554
Total Columns	1,106
Columns of type float64	636 (Continuous numeric data)
Columns of type int64	470 (Discrete numeric data)
Columns of type object	0 (Likely categorical or string)
Missing Values	1,635,818
Memory Usage	55.35 MB
<b>Target Columns Statistics</b>	
FINALWT Mean	798.62
FINALWT Standard Deviation	795.14
FINALWT Minimum	1.3263723649
FINALWT Maximum	4637.8512935
ELCNS Mean	2,720,891.26
ELCNS Standard Deviation	9,429,845.13
ELCNS Minimum	0.0
ELCNS Maximum	398,767,953.0

Table 4: Statistics of Dataset cbees12

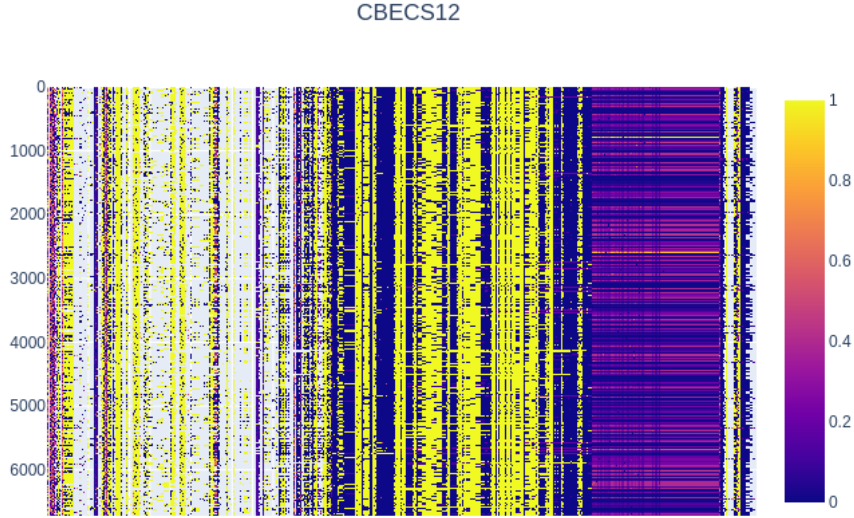


Figure 3: CBECS12 normalized data visualization

Dataset: cbecs18	Statistics
Total Entries	6,357
Total Columns	1,236
Columns of type float64	625 (Continuous numeric data)
Columns of type int64	611 (Discrete numeric data)
Columns of type object	0 (Likely categorical or string)
Missing Values	1,928,485
Memory Usage	59.99 MB
<b>Target Columns Statistics</b>	
FINALWT Mean	882.94
FINALWT Standard Deviation	1,429.63
FINALWT Minimum	1.73
FINALWT Maximum	19,028.4
ELCNS Mean	2,705,907.73
ELCNS Standard Deviation	6,425,858.09
ELCNS Minimum	36.0
ELCNS Maximum	113,727,053.0

Table 5: Statistics of Dataset cbecs18

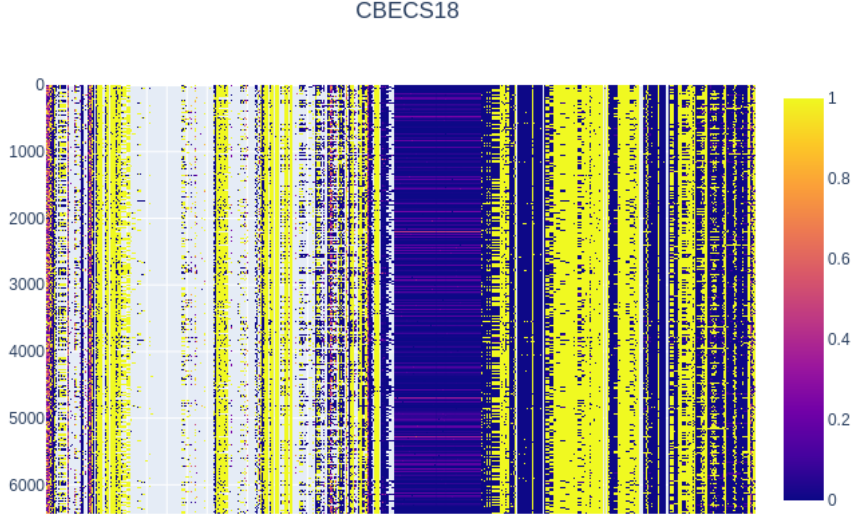


Figure 4: CBECS18 normalized data visualization

The tables and figures consistently exhibit a pattern where the majority of columns display uniform coloring, indicating that the data for each feature is highly skewed. This skewness suggests that the majority of the values for each feature are concentrated within a limited range rather than being evenly distributed. This skewness may imply that the data lacks variability, which can pose challenges for machine learning models by reducing their ability to generalize effectively. Additionally, the higher proportion of missing data in the CBECS datasets further complicates the analysis, as it necessitates careful handling, such as imputation or data removal, to prevent biased or inaccurate results. Together, these characteristics underscore the need for tailored preprocessing and feature engineering to address the data’s inherent imbalances and gaps, ensuring that any predictive models are built on a solid foundation.

Furthermore, the CBECS datasets exhibit a higher proportion of missing data compared to the RECS dataset, which could affect the overall reliability and predictive power of models trained on these datasets. Missing data can lead to biased results or reduced model accuracy if not addressed properly. In our experiments, we addressed this issue by filling the missing data with a small value of

$$1 \times 10^{-10}$$

to avoid division by zero for simplicity, as all the data are standardized before being used for training and testing. This approach ensures that the imputed values are close to the dataset’s mean after standardization, minimizing the potential distortion of the underlying data distribution while maintaining computational efficiency.



### 3.2 Research Methods

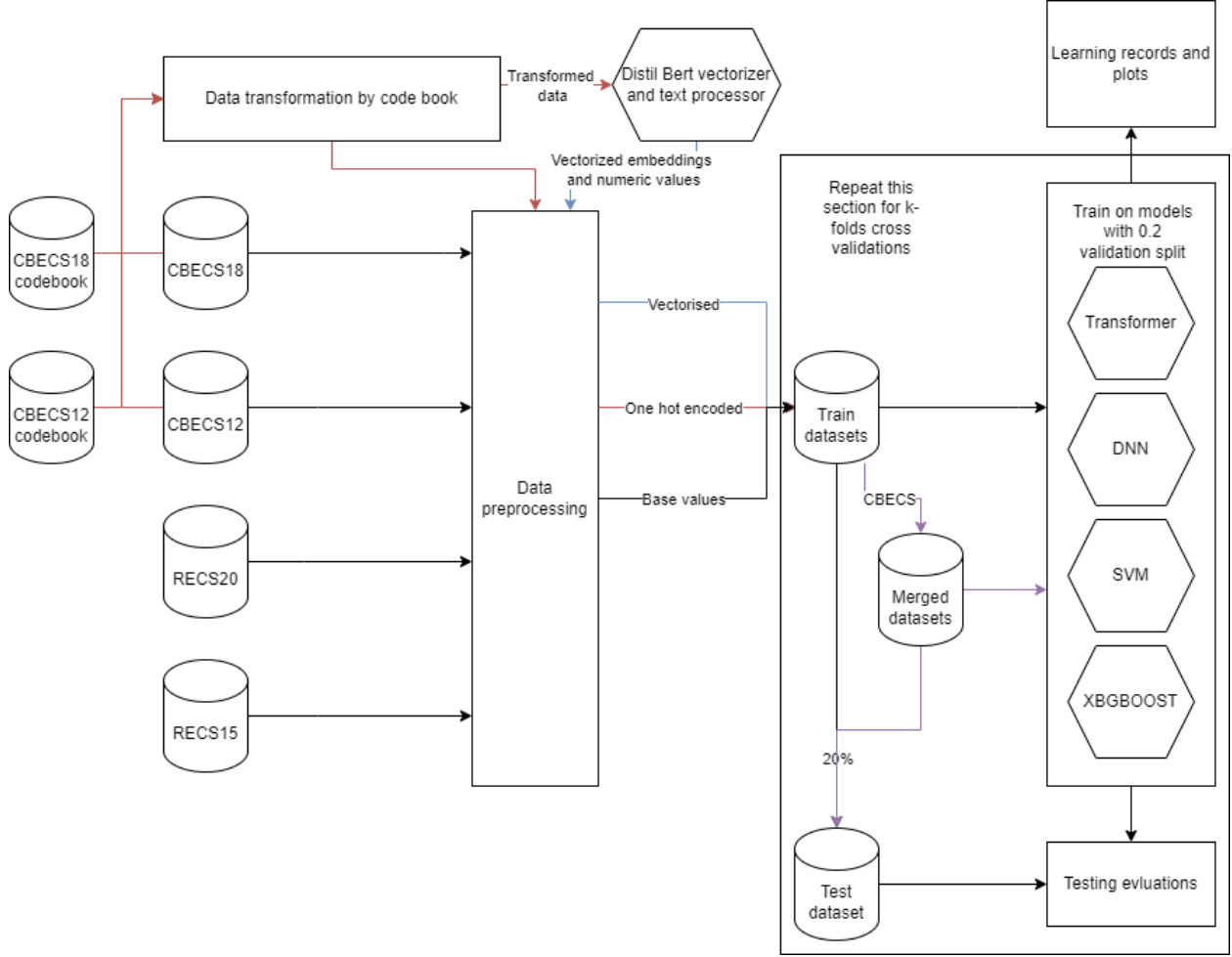


Figure 5: System architectures

The system begins with six datasets: raw data from CBECS 2018 and 2012 (CBECS18, CBECS12), their corresponding codebooks, and raw data from RECS 2020 and 2015 (RECS20, RECS15). Each raw dataset undergoes standard preprocessing to clean and prepare the data, transforming it into training datasets. These training datasets are further split into test datasets for evaluation purposes. Four models are then applied to the data: Transformer, MLP, SVM, and XGBoost.

For the Transformer and MLP models, the training data is further divided using a validation split of 0.2, meaning that 80% is used for training and 20% for validation. This validation process occurs during training and helps assess the model's performance at each epoch, with the results recorded by a learning recorder that produces performance plots over time. Once trained, all four models are evaluated using the test dataset, and the final results are output for comparison and assessment. This setup ensures that the models are trained effectively while providing clear metrics to gauge their performance on unseen data. The



reasoning behind the validation split and recorded metrics is to monitor the models’ learning progression, that we can later comment on in terms model stability and rate of convergence, allowing for adjustments based on real-time performance.

The models are later evaluated with the test dataset to produce tables of results for evaluating the generalization performance of the models. The test set provides an unbiased evaluation of how well the model performs on new, unseen data under realistic settings. Therefore, it acts as the ultimate benchmark to all comparisons between the models. This part then undergoes k-folds cross validation to provide a range of results.

For merging operations, the system first creates a copy of the original dataset and applies two merging methods. The first method concatenates the datasets based only on common features, discarding any uncommon ones. The second method retains the uncommon columns by padding placeholders for the datasets that lack those features. Once the merging is complete, the resulting datasets undergo the same process as the original training datasets. They are split into training and test sets, with the training set used to train the models and then evaluated in the same manner as before. This approach ensures that merged datasets are prepared consistently for comparison with other models.

These benchmarks establish a consistent standard for comparing more advanced models. For the data transformation comparisons, the transformation processes are carried out before any preprocessing and are aligned with the corresponding codebooks for the two CBECS datasets. This ensures that the data is converted to its correct values, particularly for categorical data that may have been previously treated as numerical. The transformation assigns the appropriate text labels to these categorical fields, reflecting their true meaning. After transformation, the data undergoes two different methods of processing, resulting in two groups: one where the text data is one-hot encoded, and another where it is transformed into embeddings using DistilBERT. This distinction allows for a detailed comparison of how different text processing methods impact model performance, as no prior studies have conducted similar operations for direct comparison. This approach allows us to assess the performance of our models in a structured way, despite the absence of exact counterparts in existing literature. Then they are sent back into the data preprocessing and go through the same process as the base datasets.

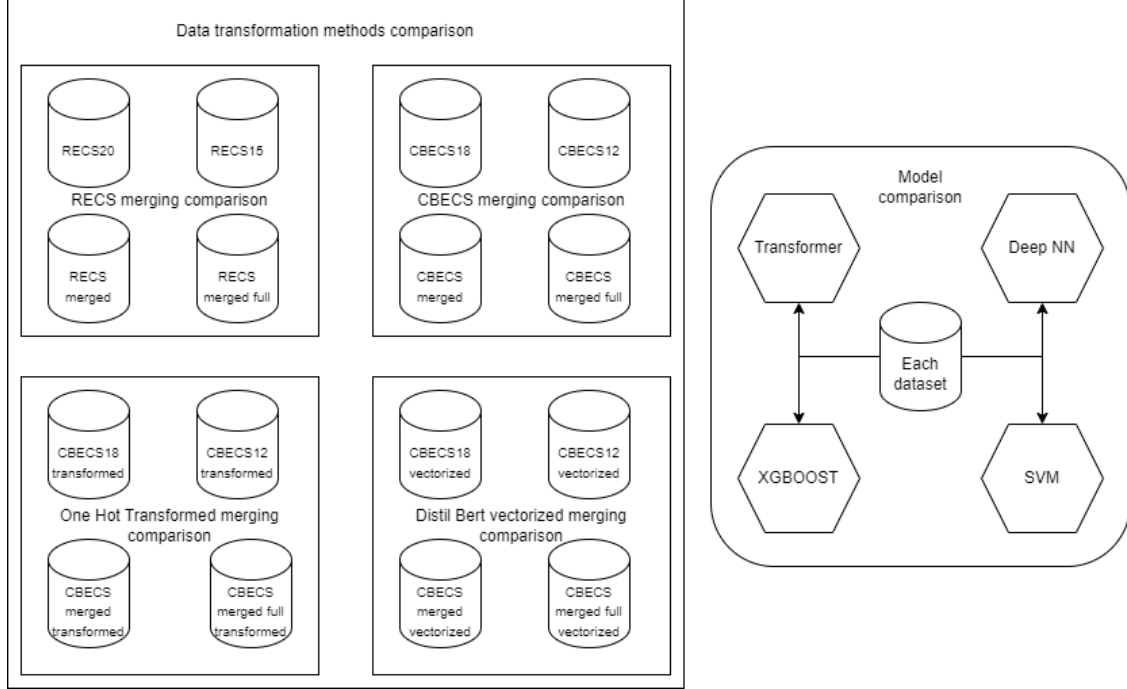


Figure 6: All datasets and models

The diagram illustrates the 16 datasets prepared for training and how they are grouped for analysis. To achieve our goals, we structure our experiments into three key types of comparisons. First, we conduct a model comparison for each dataset using four different models—Transformer, MLP, SVM, and XGBoost—to assess their performance on various data formats. This helps us understand how different machine learning approaches handle the same dataset and allows for a direct comparison of model effectiveness. We also benchmark the performance of our models against existing models from the current literature. This comparison ensures that our models are correctly implemented and validates their effectiveness. By evaluating our models relative to established models, we can demonstrate that they perform similarly or, in some cases, outperform those in prior studies. This step is crucial not only for confirming that our models are functioning as intended but also for showcasing their competitiveness and potential improvements over existing approaches in the field. These benchmarks establish a consistent standard for comparing more advanced models that employ different methods, as no prior studies have conducted similar operations for direct comparison. This approach allows us to assess the performance of our models in a structured way, despite the absence of exact counterparts in existing literature.

Second, we perform a merged data comparison using two distinct methods for combining datasets: merging by common columns, where only the shared features between datasets are retained, and merging in full by padding missing values in uncommon columns. This experiment helps determine whether merging techniques significantly affect model performance, particularly when combining datasets with varying feature spaces. These are then compared with the base models to be evaluated in their effectiveness.

Finally, we compare the effects of data transformation, this is only done to the CBECS datasets as RECS doesn't have sufficient categorical text features for this to be influential. Specifically, we evaluate untransformed datasets against those that have been correctly transformed, including one-hot encoding for categorical data and text data transformed into embeddings using DistilBERT. This comparison is crucial for understanding how feature engineering and advanced NLP techniques, such as embedding generation, impact the overall performance of the models. By exploring these dimensions, we aim to identify the most effective models, data merging strategies, and transformation methods for optimal performance. This is compared collectively as a group to other type of data transformations.

### 3.3 Data preprocessing

All data is preprocessed through standardization using `sklearn`'s `StandardScaler`, and the dataset is randomly split into training and testing sets with a ratio of 8:2 using the `train_test_split` function. During model training, the training data is further split into training and validation sets by specifying a validation split parameter, with a ratio of 0.8 to 0.2, resulting in final proportions of 6.4:1.6:2 for training, validation, and testing. Due to the absence of crucial information, the RECS dataset cannot undergo further preprocessing procedures.

#### 3.3.1 Data transformation

The CBECS dataset provides a detailed codebook, which can be utilized to transform numeric data back to its original textual form. The transformation is applied as follows: columns containing a range of values are retained as numeric, while columns with binary options, such as yes/no or yes/no/missing, are converted into their corresponding labels. For a value of 'no', the label is prepended with "NO". Any other columns representing categorical data are reverted to their original text. The transformed data is then processed in two ways: first, categorical text is converted into a neural network learnable format using one-hot encoding; second, the text is tokenized and vectorized using the DistilBERT tokenizer and model before being passed into the learning model.

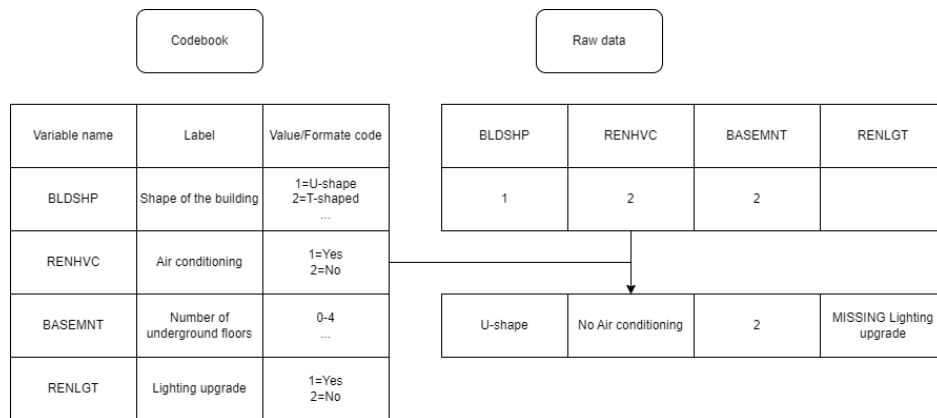
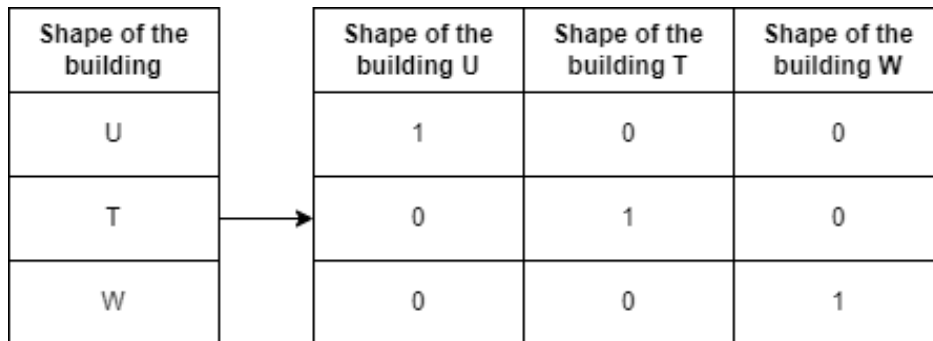


Figure 7: Data transformation by codebook

### 3.3.2 One hot

One-hot encoding is a technique used to convert categorical data into a binary matrix representation. For a categorical variable with  $N$  possible categories, one-hot encoding transforms the data into  $N$  binary features. Each feature represents a category, and for a given data point, only the binary feature corresponding to its category is set to 1, while all others are set to 0. For example, if there are three categories: **U-shape**, **T-shape**, and **W-shape**, the one-hot encoded vector for **T-shape** would be  $[0, 1, 0]$ , where each position in the vector corresponds to one of the categories.



Shape of the building	Shape of the building U	Shape of the building T	Shape of the building W
U	1	0	0
T	0	1	0
W	0	0	1

Figure 8: One hot transformation

### 3.3.3 Distil Bert vectorizer

DistilBERT is a smaller, faster, and more efficient version of BERT (Bidirectional Encoder Representations from Transformers), which was developed using a technique called knowledge distillation. It retains 97% of BERT’s performance while being 60% faster and having 40% fewer parameters. The architecture of DistilBERT mirrors that of BERT, utilizing a transformer-based encoder to learn bidirectional contextual representations from text, but with reduced layers to enhance computational efficiency. Despite its reduced size, DistilBERT remains highly effective for a wide range of natural language processing tasks, such as text classification, question answering, and token-level predictions. In our case, we use DistilBERT to convert each of the text columns into vectors to be trained before standardization and splitting. [25]

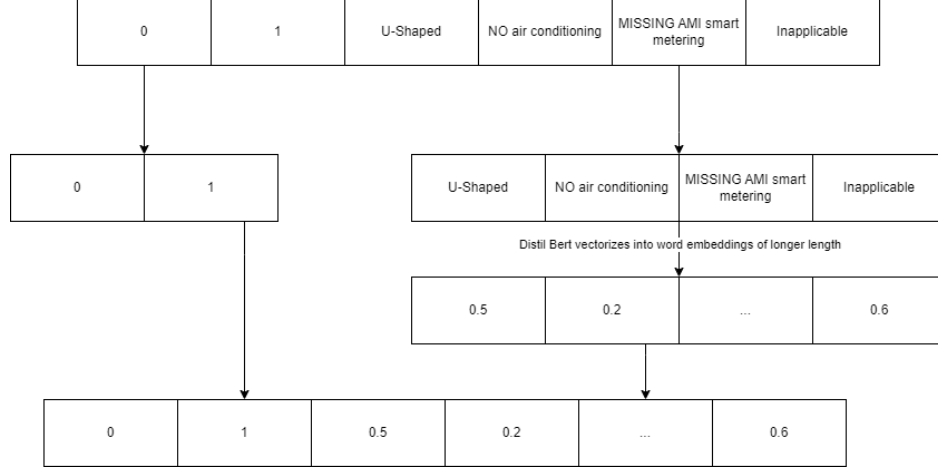


Figure 9: Distil Bert vectorizes text data

### 3.4 Data merging

The data is merged through two distinct methods. First, the datasets are merged by identifying common columns between them. In this approach, only the columns that are present in both datasets are retained, ensuring that the merged data contains consistent features shared across the datasets. Second, a full merge is performed, where all columns from both datasets are preserved. In cases where a dataset does not contain a specific column, the missing values are padded accordingly. This ensures that even the columns unique to one dataset are included in the final merged dataset, while columns that are absent from the other dataset are filled with placeholders epsilons, to maintain the structure and integrity of the data. We also make sure that the targets for  $y$  exist on both datasets, and only take the common targets as the targets for training.

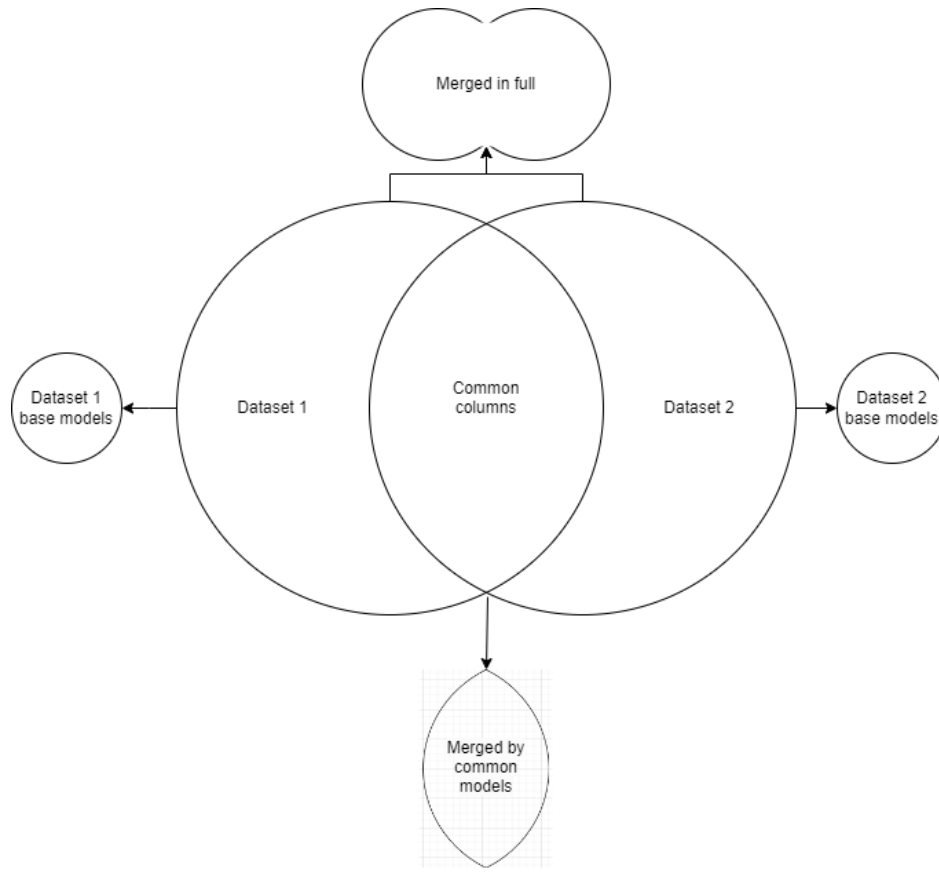


Figure 10: merging

### 3.4.1 Merging by common columns

Merging by common columns is straight forward, we first find the intersection of common column names, and concatenate the data frames based on the list of common columns.

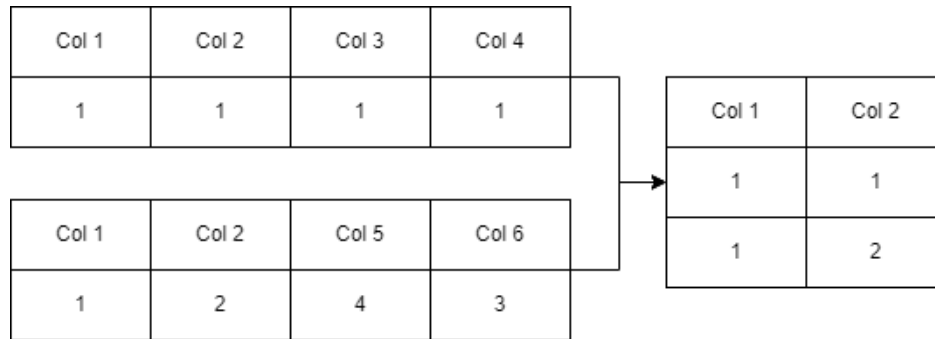


Figure 11: merge by common

### 3.4.2 Merging in full

To utilize uncommon columns that might also provide information, we pad columns that exist in one dataset but not other with missing values that is handled in other part of

reprocessing.

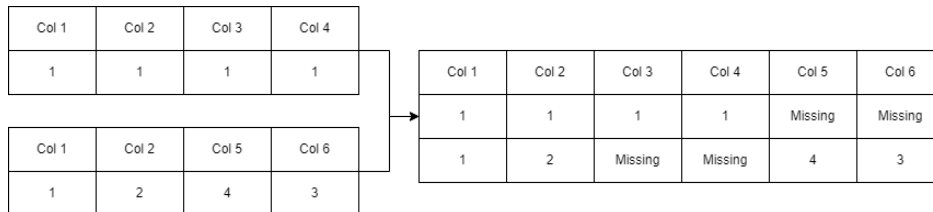


Figure 12: Full merge padding

### 3.5 Inputs and Integrity

To ensure the practical applicability of our models, we prioritize the use of realistic input data and establish a robust testing environment. Specifically, we avoid scenarios where the model is tasked with predicting the total energy consumption when all individual components of a sub-section’s energy usage are already provided as input. Such a task is inherently unrealistic because it reduces the model’s objective to performing a simple arithmetic sum, thereby eliminating the need for the model to discern meaningful patterns or relationships within the data. By designing inputs that reflect real-world complexities, we enable the model to learn and generalize effectively, ensuring its utility in practical applications.

Additionally, data leakage poses a significant challenge when merging datasets, as it can inadvertently introduce information that allows the model to access target variables during training, leading to overly optimistic performance estimates. To mitigate this risk, we implement stringent data handling procedures that ensure a clear separation between training and testing data. This involves carefully aligning and validating the merged datasets to prevent overlapping information that could compromise the model’s integrity. By addressing data leakage proactively, we maintain the reliability of our testing environment, ensuring that the model’s performance metrics accurately reflect its ability to generalize to unseen data and operate effectively in real-world scenarios.

#### 3.5.1 Input validation

To validate input columns against realistic inputs, we performed manual selection using the codebook. Apart from the item ideas, the following columns are dropped from each dataset before any other procedures are performed (refer to the appendix 10.2). The provided columns display energy usage measured in BTUs instead of the kilowatt-hours (KWH) that we are aiming to predict, along with the corresponding costs for each energy component. It’s important to highlight that accurately determining costs in advance is not possible without actual energy consumption data, since costs are inherently tied to the amount of energy used. Without knowing the specific energy usage, any cost estimates would be speculative and potentially misleading. Therefore, focusing on predicting KWH will allow for a more precise calculation of energy costs based on real usage patterns.

For CBECS, all parts related to energy consumption in BTU are dropped (refer to the appendix [10.2](#))

### 3.5.2 Data leakage

Data leakage occurs when information from the test set inadvertently influences the training process, leading to overly optimistic performance metrics that do not generalize to new data [26]. For example, if the same building appears in both CBECS datasets and is included in both the training and testing sets, and the building statistics are highly similar, the testing environment becomes compromised because the test example has effectively been used during training. To avoid this, we meticulously ensured that the same building did not appear in both the training and testing datasets simultaneously. This was crucial because having identical entities in both sets could result in the model "learning" from data it is supposed to predict, thus compromising the model's validity and the integrity of our evaluation metrics.

We conducted an integrity verification between the two merged datasets by examining the similarity of rows across common columns. Specifically, we aimed to identify any pairs of rows with a similarity exceeding 90%. The similarity between two rows  $\mathbf{r}_i$  and  $\mathbf{r}_j$  is calculated using the following formula:

$$S(\mathbf{r}_i, \mathbf{r}_j) = \frac{|\{c \in C \mid \mathbf{r}_i(c) = \mathbf{r}_j(c)\}|}{|C|} \times 100\%$$

where  $C$  represents the set of common columns between the datasets, and  $\mathbf{r}_i(c)$  denotes the value of column  $c$  in row  $\mathbf{r}_i$ . Through this analysis, no row pairs were found with a similarity score above 90%, indicating a lack of near-duplicate entries.



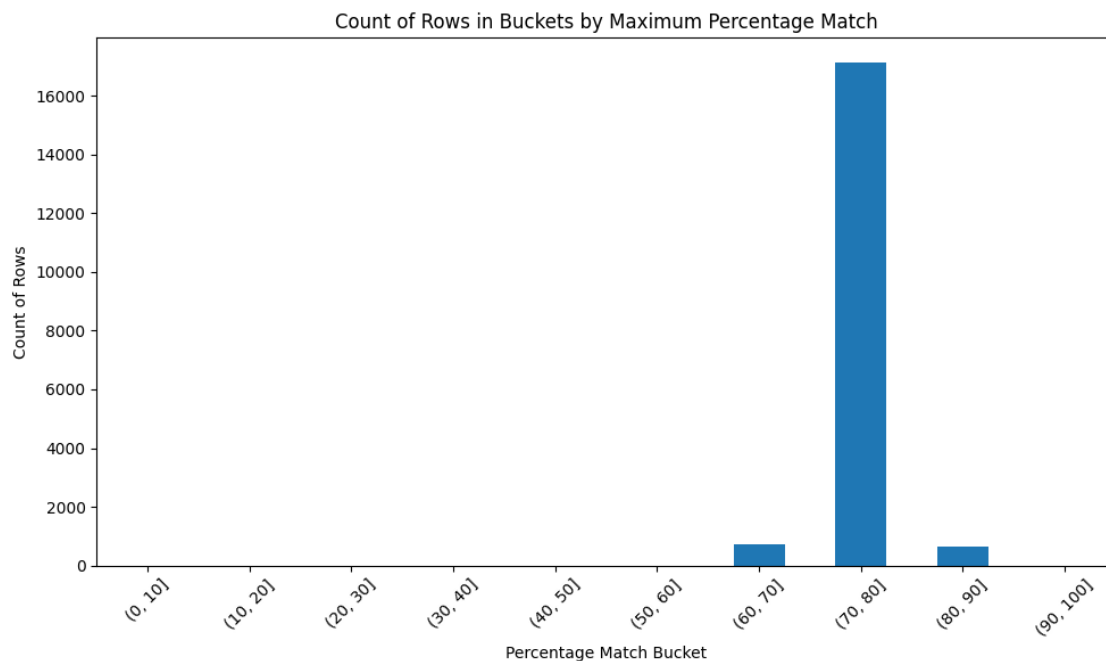


Figure 13: RECS data leakage check

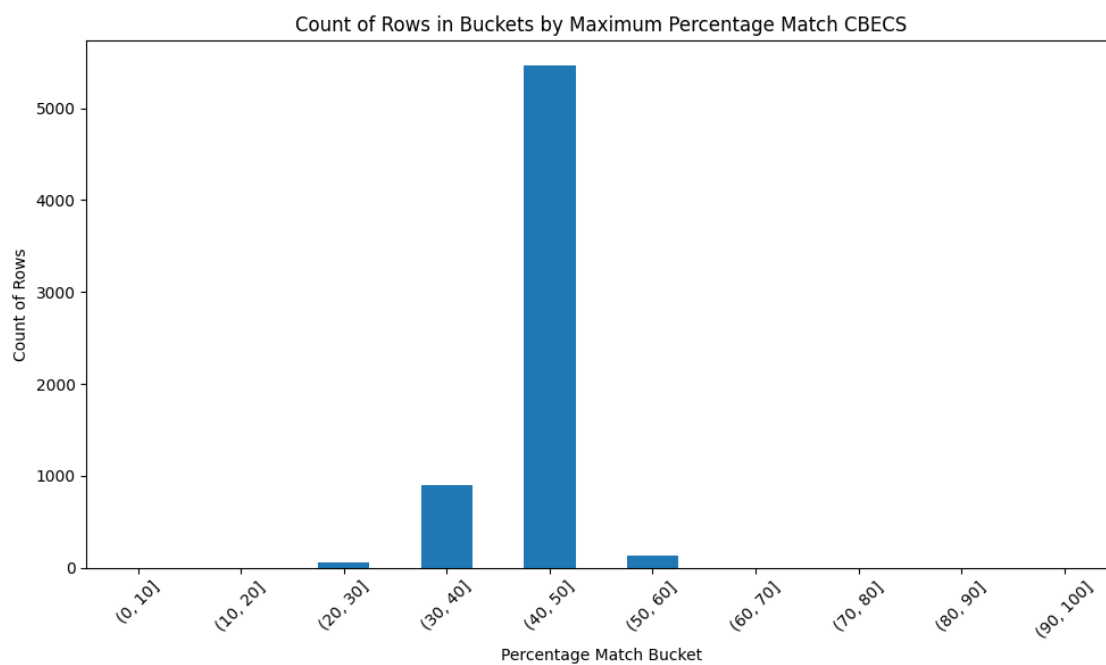


Figure 14: CBECS data leakage check

The above Figures illustrate the maximum column matching percentages for each row when comparing the datasets CBECS12 to CBECS18 and RECS15 to RECS20, respectively. In essence, for each row in one dataset, we identify the row in the corresponding dataset with

the highest column matching percentage and categorize them into distinct buckets based on this metric. The visualizations demonstrate that the majority of RECS15 rows achieve a maximum similarity of only 70% to 80% with rows in RECS20, while CBECS12 rows exhibit a maximum similarity range of 40% to 50% when compared to CBECS20 rows.

This distribution of similarity scores reinforces the integrity of our testing environment by ensuring that the merged datasets do not contain highly similar or duplicate rows. The absence of high similarity scores minimizes the risk of biased or skewed results due to redundant data, thereby maintaining the robustness and reliability of our analytical processes. Although this test does not entirely guarantee that we will avoid data leakage problems, it does ensure our model's integrity. For data leakage to have a significant effect, the model would need to encounter very similar examples, and our verification has confirmed that this is not the case.

## 3.6 Models

### 3.6.1 Model Architectures

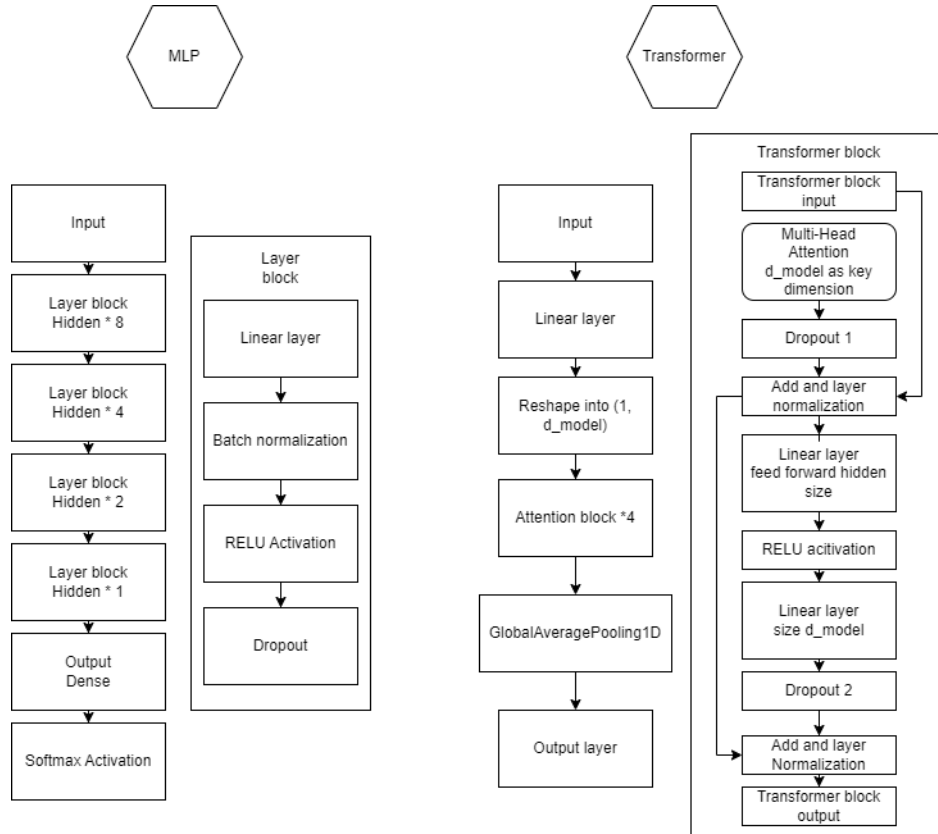


Figure 15: MLP and Transformer architecture

The figure above illustrates the architecture of two deep learning models. The MLP model is designed for simplicity, incorporating standard enhancements like batch normalization and

dropout to improve stability and reduce overfitting. It also includes a down sampling layer to assist with handling high-dimensional input data, ensuring the model remains efficient while preventing overfitting. These additions contribute to better generalization and performance, especially when working with large datasets or complex input features (Liu, J. et al) [27]. The Transformer model is adopted with minimal changes from the original by Vaswani, A. [28]. It features only the encoder and ignores positional encodings which does not apply to our data.

### 3.6.2 Transformer

The transformer model presented consists of multiple stacked transformer blocks, each encapsulating key mechanisms such as multi-head attention, layer normalization, and feedforward neural networks, all integrated with residual connections and dropout regularization. At the core of each `TransformerBlock`, the multi-head attention mechanism allows the model to focus on different representation subspaces by linearly projecting the input into queries  $Q$ , keys  $K$ , and values  $V$ , and computing attention scores across multiple heads. Mathematically, the scaled dot-product attention for a single head is defined as [28]:

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^\top}{\sqrt{d_k}} \right) V$$

where  $d_k$  is the dimensionality of the keys. In the multi-head context, this operation is performed in parallel across  $h$  heads, and the results are concatenated and linearly transformed to produce the final attention output. Following the attention layer, layer normalization is applied to stabilize and accelerate the training process. The layer normalization operation for an input  $x$  is given by [28]:

$$\text{LayerNorm}(x) = \frac{x - \mu}{\sigma + \epsilon} \odot \gamma + \beta$$

where  $\mu$  and  $\sigma$  are the mean and standard deviation of  $x$ , respectively, and  $\gamma$  and  $\beta$  are learnable parameters that scale and shift the normalized output. Residual connections are incorporated by adding the input of the attention layer to its output, followed by layer normalization, ensuring gradient flow and mitigating vanishing gradients.

Subsequently, the model employs a position-wise feedforward network composed of two dense layers with a ReLU activation in between, transforming the normalized output. This feedforward component can be expressed as [28]:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

where  $W_1$  and  $W_2$  are weight matrices, and  $b_1$  and  $b_2$  are bias vectors. Another residual connection is applied after the feedforward network, followed by a second layer normalization. Dropout layers are interspersed after both the attention and feedforward outputs to prevent overfitting by randomly setting a fraction of the input units to zero during training.

The overall transformer model begins by projecting the input into a higher-dimensional space using a dense layer, reshaped to form a sequence suitable for the transformer blocks. Multiple `TransformerBlock` instances are then stacked, allowing the model to capture complex hierarchical representations. After the transformer layers, a global average pooling layer aggregates the sequence information, and a final dense layer maps the pooled features to the desired output shape.

### 3.6.3 MLP

The linear MLP model consists of a sequence of dense (fully connected) layers, each followed by batch normalization, a ReLU activation function, and dropout regularization. This architecture is designed to facilitate efficient training and enhance the model’s generalization capabilities. Each dense layer performs a linear transformation of the input data, defined mathematically as [29]:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

where  $\mathbf{W}$  is the weight matrix,  $\mathbf{x}$  is the input vector, and  $\mathbf{b}$  is the bias vector. In the given model, the number of units in each dense layer decreases progressively (e.g., from  $8 \times \text{dim}$  to  $2 \times \text{dim}$ ), allowing the model to learn hierarchical feature representations.

Following each dense layer, batch normalization is applied to stabilize and accelerate the training process. Batch normalization normalizes the output of the previous layer by adjusting and scaling the activations. The operation is defined as [29]:

$$\text{BatchNorm}(\mathbf{x}) = \gamma \frac{\mathbf{x} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} + \beta$$

where  $\mu_{\mathcal{B}}$  and  $\sigma_{\mathcal{B}}^2$  are the mean and variance computed over the current mini-batch  $\mathcal{B}$ ,  $\gamma$  and  $\beta$  are learnable parameters that scale and shift the normalized output, and  $\epsilon$  is a small constant added for numerical stability.

To prevent overfitting and enhance the model’s ability to generalize to unseen data, dropout regularization is employed after the activation function. Dropout works by randomly setting a fraction of the input units to zero during training, effectively preventing the network from becoming overly reliant on specific neurons. Mathematically, if  $p$  is the dropout rate, the dropout operation can be represented as [30]:

$$\text{Dropout}(\mathbf{x}) = \mathbf{x} \odot \mathbf{m}$$

where  $\mathbf{m}$  is a binary mask with each element independently set to 1 with probability  $1 - p$  and 0 with probability  $p$ , and  $\odot$  denotes element-wise multiplication. This stochastic regularization technique encourages the network to develop redundant representations, thereby improving its robustness. This differs from the implementation of the FFN in transformer model, as batch normalization is applied before the activation.

In summary, this linear model leverages batch normalization and ReLU activations to facilitate stable and efficient training while employing dropout to enhance generalization. The combination of these techniques results in a robust architecture capable of learning complex mappings from input to output.

### 3.6.4 Output and compile

The final layer of the models is a dense layer with a linear activation function, which maps the learned features to the desired output shape. The model is compiled using the Adam optimizer, known for its adaptive learning rate capabilities, which combines the advantages of both AdaGrad and RMSProp optimizers. The optimization process minimizes the mean squared error (MSE) loss function, defined as [31]:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

where  $y_i$  are the true values and  $\hat{y}_i$  are the predicted values. Additionally, the model tracks the mean absolute error (MAE) and the coefficient of determination (R-squared) as performance metrics to evaluate its predictive performance and explanatory power.

For optimization, the models employs the Adam optimizer, which adapts the learning rate for each parameter based on estimates of first and second moments of the gradients. The update rules for Adam are defined as [32]:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \\ \theta_t &= \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned}$$

where  $g_t$  is the gradient at time step  $t$ ,  $m_t$  and  $v_t$  are the first and second moment estimates,  $\beta_1$  and  $\beta_2$  are decay rates,  $\alpha$  is the learning rate, and  $\epsilon$  is a small constant to prevent division by zero.

The targets for the RECS models are all the electricity usages in KWH refer to the appendix 10.3

The targets for CBECS models are FINALWT and ELCNS as shown in the data guide book [23] with weight of the building and total energy consumption. These values are also standardised such that MAE in testing results are all consistent and comparable with each other in other datasets. We selected R-squared as our primary metric along with MAE because the models are performing regression tasks, where accuracy is not an appropriate measure.

## 4 Results

### 4.1 R-squared Testing Results

MODEL	RECS15	RECS20	RECS merged	RECS merged in full
Transformer	<b>0.7390 <math>\pm</math> 0.0035</b>	<b>0.8197 <math>\pm</math> 0.0023</b>	<b>0.6824 <math>\pm</math> 0.0058</b>	<b>0.7823 <math>\pm</math> 0.0054</b>
MLP	0.6305 $\pm$ 0.0030	0.7440 $\pm$ 0.0091	0.6593 $\pm$ 0.0046	0.7241 $\pm$ 0.0115
XGBOOST	0.0005 $\pm$ 0.3688	0.2245 $\pm$ 0.1574	0.0717 $\pm$ 0.0452	0.4131 $\pm$ 0.0426
SVM	0.0168 $\pm$ 0.0716	0.0361 $\pm$ 0.0280	0.0383 $\pm$ 0.0278	0.0700 $\pm$ 0.0085
CBECS	CBECS12	CBECS18	CBECS merged	CBECS merged in full
Transformer	0.9258 $\pm$ 0.0638	0.9402 $\pm$ 0.0193	<b>0.9471 <math>\pm</math> 0.0482</b>	0.9437 $\pm$ 0.0362
MLP	0.8122 $\pm$ 0.0530	0.8527 $\pm$ 0.0217	0.8934 $\pm$ 0.0414	0.8793 $\pm$ 0.0374
XGBOOST	0.7578 $\pm$ 0.1421	0.9056 $\pm$ 0.0347	0.8353 $\pm$ 0.0920	0.8364 $\pm$ 0.1056
SVM	0.5263 $\pm$ 0.2011	0.6582 $\pm$ 0.0835	0.6250 $\pm$ 0.1379	0.6259 $\pm$ 0.1476
CBECS One Hot transformed				
Transformer	0.5002 $\pm$ 0.0627	0.5181 $\pm$ 0.0520	0.5245 $\pm$ 0.0513	0.9452 $\pm$ 0.0368
MLP	0.4533 $\pm$ 0.0548	0.4398 $\pm$ 0.0368	0.5032 $\pm$ 0.0412	0.8773 $\pm$ 0.0274
XGBOOST	0.5334 $\pm$ 0.0632	0.5105 $\pm$ 0.0280	0.4843 $\pm$ 0.0903	0.8364 $\pm$ 0.1056
SVM	0.3817 $\pm$ 0.1205	0.3507 $\pm$ 0.0580	0.4290 $\pm$ 0.0947	0.6259 $\pm$ 0.1476
CBECS DistilBERT embedding transformed				
Transformer	<b>0.9641 <math>\pm</math> 0.0010</b>	<b>0.9712 <math>\pm</math> 0.0193</b>	0.9447 $\pm$ 0.0428	<b>0.9705 <math>\pm</math> 0.0385</b>
MLP	0.8759 $\pm$ 0.0032	0.8705 $\pm$ 0.0184	0.8665 $\pm$ 0.0252	0.8827 $\pm$ 0.0431
XGBOOST	0.0793 $\pm$ 0.8683	<b>0.9504 <math>\pm</math> 0.0285</b>	0.7026 $\pm$ 0.3083	0.8072 $\pm$ 0.1116
SVM	0.5700 $\pm$ 0.1537	0.7763 $\pm$ 0.0525	0.7905 $\pm$ 0.1701	0.8814 $\pm$ 0.1652

Table 6: Performance  $R^2$

#### 4.1.1 Models comparison

The table presents the testing results using R-squared for each model across 5-fold cross-validation, with key models highlighted in bold. Compared to the previous study, where XGBoost achieved an R-squared of 0.6554 [20] for the RECS15 dataset, our XGBoost result of  $0.0005 \pm 0.3688$  performed much worse. On the CBECS12 dataset, previous studies reported the best-performing model as XGBoost, achieving a score of  $0.89 \pm 0.01$  [17] when validated on the City Law dataset. Although we did not perform the same validation and

feature selection whereas the original paper has 19 continuous variables and 56 categorical variables selected as inputs, our XGBoost model obtained a score of  $0.7578 \pm 0.1421$ , which is suboptimal. In contrast, our transformer model achieved a higher score of  $0.9258 \pm 0.0638$ . This may be attributed to the lack of extensive feature selection, resulting in a dimensionality too high for XGBoost to generalize effectively. However, our transformer models outperformed their best XGBoost model both the RECS15 and CBECS12 dataset, providing a benchmark for comparison with other RECS and CBECS models.

For the **RECS15 dataset**, the Transformer model achieved the highest R-squared value of  $0.7390 \pm 0.0035$ , indicating a strong ability to predict with minimal variance. The MLP model followed with  $0.6305 \pm 0.0030$ , still performing well but clearly behind the Transformer. XGBoost, however, showed a dramatic drop in performance with an R-squared of only  $0.0005 \pm 0.3688$ , suggesting that it struggled significantly with this dataset. Similarly, SVM had a very low R-squared of  $0.0168 \pm 0.0716$ , performing only marginally better than XGBoost but still far below the deep learning models.

For the **RECS20 dataset**, the Transformer again led with an R-squared of  $0.8197 \pm 0.0023$ , outperforming all other models with minimal variance. The MLP also performed well, achieving  $0.7440 \pm 0.0091$ , though with a slightly higher variance. XGBoost saw some improvement in this dataset, with an R-squared of  $0.2245 \pm 0.1574$ , but it was still far behind the deep learning models. SVM continued to perform poorly, with an R-squared of only  $0.0361 \pm 0.0280$ .

In the **RECS merged dataset**, the Transformer model achieved an R-squared of  $0.6824 \pm 0.0058$ , maintaining its lead, while the MLP was close behind with  $0.6593 \pm 0.0046$ . Both models performed reasonably well on the merged data. XGBoost and SVM, however, continued to struggle, with XGBoost achieving only  $0.0717 \pm 0.0452$  and SVM similarly low at  $0.0383 \pm 0.0278$ .

For the **RECS merged in full dataset**, the Transformer once again performed the best with an R-squared of  $0.7823 \pm 0.0054$ , further solidifying its dominance across datasets. The MLP followed with  $0.7241 \pm 0.0115$ , maintaining respectable performance. XGBoost showed some improvement on this larger merged dataset, with an R-squared of  $0.4131 \pm 0.0426$ , but still lagged far behind the deep learning models. SVM had an R-squared of  $0.0700 \pm 0.0085$ , continuing its poor performance across all datasets.

In summary, the Transformer model consistently outperformed other models across all RECS datasets, followed by the MLP, which also delivered competitive results. XGBoost and SVM, however, struggled to generalize and performed significantly worse, particularly on datasets with higher dimensionality, such as the RECS15 dataset. The lack of feature selection likely contributed to XGBoost’s poor performance, while the deep learning models managed to better handle the complex relationships within the data.

For the **CBECS12 dataset**, the Transformer model achieved an R-squared value of  $0.9258 \pm 0.0638$ , significantly outperforming all other models. The MLP followed with  $0.8122$

$\pm 0.0530$ , maintaining a competitive but clearly lower performance. XGBoost performed moderately well with  $0.7578 \pm 0.1421$ , while SVM lagged far behind with an R-squared of  $0.5263 \pm 0.2011$ .

In the **CBECS18 dataset**, the Transformer model improved further with an R-squared of  $0.9402 \pm 0.0193$ , and the MLP model also showed good results with  $0.8527 \pm 0.0217$ . XGBoost performed better in this dataset with  $0.9056 \pm 0.0347$ , narrowing the gap to the deep learning models. SVM, however, remained significantly lower with  $0.6582 \pm 0.0835$ .

For the **CBECS merged dataset**, the Transformer continued to dominate with an R-squared of  $0.9471 \pm 0.0482$ , followed by the MLP at  $0.8934 \pm 0.0414$ . XGBoost managed an R-squared of  $0.8353 \pm 0.0920$ , performing reasonably well but still behind the deep learning approaches. SVM, on the other hand, recorded an R-squared of  $0.6250 \pm 0.1379$ .

In the **CBECS merged in full dataset**, the Transformer achieved a strong performance with  $0.9437 \pm 0.0362$ , while the MLP performed well with  $0.8793 \pm 0.0374$ . XGBoost also held its ground with  $0.8364 \pm 0.1056$ , while SVM remained consistent with an R-squared of  $0.6259 \pm 0.1476$ .

For the **CBECS One Hot transformed datasets**, the Transformer results dropped slightly, achieving an R-squared of  $0.5002 \pm 0.0627$  for CBECS12 but then improved to  $0.9452 \pm 0.0368$  for the full dataset. The MLP model similarly showed lower performance, starting with  $0.4533 \pm 0.0548$  on CBECS12 and improving to  $0.8773 \pm 0.0274$  on the full dataset. XGBoost performed slightly better for CBECS12 at  $0.5334 \pm 0.0632$  but did not surpass the Transformer in the full dataset, achieving  $0.8364 \pm 0.1056$ . SVM performed the worst on the one-hot transformed data, with results ranging from  $0.3817 \pm 0.1205$  to  $0.6259 \pm 0.1476$ .

When using the **CBECS DistilBERT embedding transformed datasets**, the Transformer model showed its best performance across all datasets, achieving an impressive  $0.9641 \pm 0.0010$  for CBECS12 and maintaining strong results in all versions, reaching  $0.9705 \pm 0.0385$  for the full dataset. The MLP also performed well, ranging from  $0.8759 \pm 0.0032$  to  $0.8827 \pm 0.0431$  in the full dataset. Notably, XGBoost exhibited significantly lower performance on the CBECS12 dataset, achieving a metric of  $0.0793 \pm 0.8683$ , which may be attributed to several factors. Firstly, CBECS12 might have contained a higher level of noise or irrelevant features that hindered XGBoost’s ability to effectively capture underlying patterns. Additionally, the dataset size and feature distribution in CBECS12 could have been less conducive to XGBoost’s gradient boosting mechanisms, potentially leading to overfitting or underfitting. In contrast, the substantial improvement to  $0.9504 \pm 0.0285$  on the CBECS18 dataset suggests that the newer dataset likely had cleaner data, more relevant features, or a structure that better aligned with XGBoost’s strengths in handling complex, non-linear relationships. Moreover, advancements in feature engineering or pre-processing techniques between the two datasets might have enhanced the model’s performance on CBECS18. On the other hand, Support Vector Machines (SVM) demonstrated moderate



results across both datasets, possibly due to their inherent limitations in scaling with larger datasets or capturing intricate patterns compared to ensemble methods like XGBoost. This indicates that while SVMs are robust to certain types of data variations, they may not fully leverage the improvements present in CBECS18 as effectively as XGBoost does.

Overall, the Transformer model consistently outperformed the other models across all CBECS datasets and transformations, with the DistilBERT embedding transformation yielding the highest results. The MLP also showed competitive performance, while XGBoost and SVM lagged behind, particularly on certain dataset variations. Transformer models consistently exhibit the highest R-squared values across both RECS and CBECS datasets, including their transformed versions. This indicates a strong fit to the data. Correspondingly, Transformers maintain the lowest MAE values, reinforcing their superior predictive performance. MLPs generally perform well, ranking second only to Transformers. They demonstrate robust performance across most datasets. Their MAE values are slightly higher than those of Transformers but remain competitive, aligning with their R-squared performance. XGBOOST shows mixed performance. While it performs adequately on some datasets (e.g., CBECS20 with  $R\text{-squared} = 0.9504$ ), it exhibits extremely low R-squared values in others (e.g., CBECS12 DistilBERT transformed with  $R\text{-squared} = 0.0793 \pm 0.8683$ ).

The Transformer model is outperforming the MLP in testing primarily due to its ability to capture complex dependencies and long-range relationships between features, which is particularly important in datasets with high-dimensional inputs such as CBECS. Unlike MLPs, which rely on fully connected layers to process input data, the Transformer architecture employs self-attention mechanisms, allowing it to weigh the importance of each input feature relative to others across the entire sequence. This helps the Transformer model better handle data with intricate feature interactions or varying importance of features, which may be present in building energy consumption datasets. Additionally, the Transformer’s capacity to process inputs in parallel rather than sequentially enables more efficient learning and generalization during training, reducing overfitting and improving performance on unseen test data. Furthermore, the self-attention mechanism reduces the dependence on the number of layers required to capture complex interactions, making Transformers more effective than MLPs when dealing with large, heterogeneous datasets, as observed in the testing results.

#### 4.1.2 Merging comparison

In the **base datasets**, such as RECS15 and CBECS12, each dataset is treated independently, and the models are able to focus on specific patterns inherent to those individual datasets. The Transformer model performs well in these cases because of its capacity to capture complex relationships within the feature space and manage high-dimensional data. However, the MLP, which does not benefit from attention mechanisms, may struggle to identify intricate interactions between features, especially as these datasets are relatively small and can lead to underfitting.

When the datasets are **merged by common features**, such as the RECS and CBECS

datasets merged on overlapping feature columns, the performance generally improves for most models, particularly the Transformer and MLP. This improvement occurs because merging datasets by common features increases the dataset size, allowing models to learn more robust patterns. The merged datasets also benefit from the shared feature space, which can help models generalize better. Transformers, with their ability to handle varied feature importance and long-range dependencies, excel in this scenario by efficiently leveraging the additional data and common feature interactions. In contrast, MLPs see some improvement but may not fully capitalize on the complexity of the merged dataset due to their simpler architecture.

In the case of **merged in full datasets**, where all feature columns from both datasets are combined regardless of overlap, the model performance typically peaks for deep learning models like Transformers. This is because the full merged dataset provides a significantly larger and more diverse feature set, allowing the model to learn from a wider variety of patterns and interactions. The Transformer model’s ability to handle large-scale, high-dimensional data makes it particularly effective here, as it can focus on the most important features through its self-attention mechanism, reducing the noise from less relevant features. On the other hand, the MLP, while still benefiting from the larger dataset, may not be able to distinguish between important and irrelevant features as effectively, leading to slightly lower performance compared to the Transformer. Additionally, traditional models like XGBoost and SVM struggle with the high dimensionality and the absence of careful feature selection, which can cause overfitting or underperformance in the full merged datasets [33].

#### 4.1.3 Transformation comparison

In the **base datasets** (e.g., RECS and CBECS), the models are applied directly to the raw numerical and categorical features without additional transformations. These datasets provide a straightforward representation of the data, but the feature space is often limited, especially for categorical variables. Models like Transformers and MLPs can still perform well on these datasets because they can learn from numerical correlations and simple categorical encodings, but the potential for capturing deeper patterns is constrained. Transformers tend to outperform MLPs in these base datasets due to their ability to handle complex interactions and capture feature importance dynamically.

For the **one-hot transformed datasets**, categorical variables are converted into a binary feature space, dramatically increasing the dimensionality of the data. This transformation often improves model performance in traditional machine learning models (e.g., XGBoost, SVM) because it allows these models to treat each category as an independent feature, providing a more detailed representation of the data. However, the significant increase in dimensionality poses a challenge, especially for models like XGBoost, which may struggle to generalize in the presence of many irrelevant or redundant features. Transformers can still perform relatively well under one-hot transformations due to their ability to handle high-dimensional data through self-attention mechanisms, but the effectiveness may diminish slightly if the additional features introduce noise or irrelevant information. MLPs may suffer more from this transformation, as they are less equipped to prioritize important fea-

tures in high-dimensional spaces, leading to overfitting or suboptimal performance. Overall, the one-hot encoding transformation results in poorer testing performance compared to both using no transformation and applying the DistilBERT transformation.

The **DistilBERT-transformed datasets** represent a significant shift in feature representation, where categorical and text-based features are embedded into dense vector spaces using pre-trained language models like DistilBERT. This transformation leads to a highly expressive and compact feature representation, which retains semantic meaning and captures relationships between categories that would otherwise be missed in one-hot encoding. Transformers excel in this setting because they are well-suited to process dense embeddings and can fully exploit the rich, contextualized information provided by DistilBERT. This results in superior performance, as seen in the testing results, where Transformers consistently outperform all other models when using DistilBERT embeddings. MLPs also benefit from the compact and meaningful feature representations provided by DistilBERT but may still lag behind Transformers in terms of capturing more nuanced interactions between embedded features. XGBoost and SVM tend to struggle with DistilBERT embeddings due to their reliance on structured, tabular data and lack of inherent mechanisms to process dense, unstructured feature vectors effectively.

In summary, **base datasets** provide a simple and limited feature space, favoring models like Transformers that can handle basic interactions effectively. **One-hot transformed datasets** increase the feature space by breaking down categorical variables, which benefits some models but poses challenges for others, particularly in handling high-dimensional data. The **DistilBERT-transformed datasets**, however, offer a more sophisticated and context-rich feature representation, leading to substantial improvements in model performance, especially for Transformers, due to their ability to capture complex patterns and relationships within the dense embeddings.

## 4.2 MAE Testing Results

MODEL	RECS15	RECS20	RECS merged	RECS merged in full
Transformer	$0.3256 \pm 0.0035$	$0.2323 \pm 0.0028$	$0.2993 \pm 0.0022$	$0.2541 \pm 0.0037$
MLP	$0.4015 \pm 0.0054$	$0.2980 \pm 0.0019$	$0.3291 \pm 0.0018$	$0.3116 \pm 0.0052$
XGBOOST	$0.1827 \pm 0.0082$	$0.1993 \pm 0.0065$	$0.2662 \pm 0.0108$	$0.2105 \pm 0.0056$
SVM	$0.2546 \pm 0.0235$	$0.2351 \pm 0.0091$	$0.2496 \pm 0.0138$	$0.2986 \pm 0.0092$
CBECS	CBECS12	CBECS18	CBECS merged	CBECS merged in full
Transformer	$0.0885 \pm 0.0067$	$0.1049 \pm 0.0066$	$0.0806 \pm 0.0079$	$0.0830 \pm 0.0097$
MLP	$0.2004 \pm 0.0072$	$0.1995 \pm 0.0057$	$0.1181 \pm 0.0064$	$0.1230 \pm 0.0080$
XGBOOST	$0.0528 \pm 0.0138$	$0.0434 \pm 0.0055$	$0.0391 \pm 0.0035$	$0.0405 \pm 0.0040$
SVM	$0.1262 \pm 0.0201$	$0.1195 \pm 0.0068$	$0.0873 \pm 0.0053$	$0.0959 \pm 0.0063$
CBECS One Hot transformed				
Transformer	$0.3645 \pm 0.0137$	$0.3094 \pm 0.0121$	$0.2944 \pm 0.0133$	$0.0881 \pm 0.0139$
MLP	$0.4253 \pm 0.0073$	$0.4155 \pm 0.0066$	$0.3438 \pm 0.0057$	$0.1286 \pm 0.0089$
XGBOOST	$0.2400 \pm 0.0179$	$0.2213 \pm 0.0065$	$0.2055 \pm 0.0030$	$0.0405 \pm 0.0040$
SVM	$0.2538 \pm 0.0218$	$0.2508 \pm 0.0098$	$0.2093 \pm 0.0049$	$0.0959 \pm 0.0063$
CBECS DistilBERT embedding transformed				
Transformer	$0.1327 \pm 0.0025$	$0.0935 \pm 0.0167$	$0.1049 \pm 0.0266$	$0.0671 \pm 0.0059$
MLP	$0.2619 \pm 0.0014$	$0.2027 \pm 0.0072$	$0.1661 \pm 0.0053$	$0.1237 \pm 0.0093$
XGBOOST	$0.0444 \pm 0.0127$	$0.0184 \pm 0.0029$	$0.0266 \pm 0.0061$	$0.0238 \pm 0.0057$
SVM	$0.1655 \pm 0.0178$	$0.1180 \pm 0.0041$	$0.0975 \pm 0.0059$	$0.0515 \pm 0.0041$

Table 7: Performance *MAE*

For most models and datasets, a coherent relationship exists where higher R-squared values correspond to lower Mean Absolute Error (MAE), indicating consistent performance evaluations. However, anomalies such as the performance of XGBoost on the CBECS DistilBERT Embedding Transformed (CBECS12) subset—characterized by a low R-squared alongside a low MAE—are atypical. This discrepancy suggests that while the model accurately predicts values close to the actual measurements on average (resulting in a low MAE), it fails to capture the underlying variance or trends within the data, as evidenced by the low R-squared.

In contrast, the "merged in full" subset demonstrates a high R-squared coupled with a relatively low MAE for XGBoost, indicating that the model not only fits the data well in

terms of variance explanation but also maintains low absolute prediction errors. This alignment is expected and reinforces the model’s reliability on comprehensive datasets. However, a decline in R-squared across other subsets, along with corresponding fluctuations in MAE, highlights inconsistent model behavior when applied to different data partitions. Typically, XGBoost achieves lower MAE than Support Vector Machines (SVM), but it does not consistently match the superior MAE performance of Transformer-based models or Multi-Layer Perceptrons (MLPs).

Notably, SVMs generally record the lowest R-squared values across all models, indicating weaker explanatory power, and exhibit higher MAE values compared to Transformers and MLPs. Specifically, XGBoost exhibits an extremely low R-squared value of 0.0793 with a high variance ( $\pm 0.8683$ ) on the CBECS12 subset, suggesting highly unstable performance for this particular dataset. Interestingly, the MAE remains relatively low ( $0.0444 \pm 0.0127$ ), which is inconsistent with the poor R-squared value. Typically, a low MAE correlates with a higher R-squared, implying that this discrepancy may result from issues such as overfitting, data leakage, or anomalies within the dataset.

Deep learning models, particularly Transformers and MLPs, demonstrate superior and consistent performance across most datasets, as evidenced by their high R-squared and low MAE values. Transformers leverage their attention mechanisms to capture complex patterns and dependencies in the data, making them highly effective for various feature representations, including DistilBERT embeddings. MLPs, with their layered architecture, excel in modeling non-linear relationships, contributing to their robust performance metrics.

The varying performance of Transformer models across different feature transformations—such as One-Hot encoding versus DistilBERT embeddings—indicates that the choice of feature representation significantly influences model efficacy. This highlights the importance of selecting appropriate embedding techniques to harness the full potential of deep learning architectures. Exploring hybrid approaches or alternative embedding strategies may further enhance model consistency and performance.

Given their demonstrated effectiveness, Transformers and MLPs should be prioritized for deployment in practical applications. Nonetheless, understanding the specific contexts or data characteristics where other models like XGBoost excel can provide valuable flexibility and robustness to the modeling strategy. Incorporating additional evaluation metrics, such as Root Mean Squared Error (RMSE), MAE distribution analysis, and residual diagnostics, can offer a more comprehensive understanding of model performance, particularly in cases where R-squared and MAE provide conflicting insights.

### 4.3 Transformer and MLP Comparison During Training

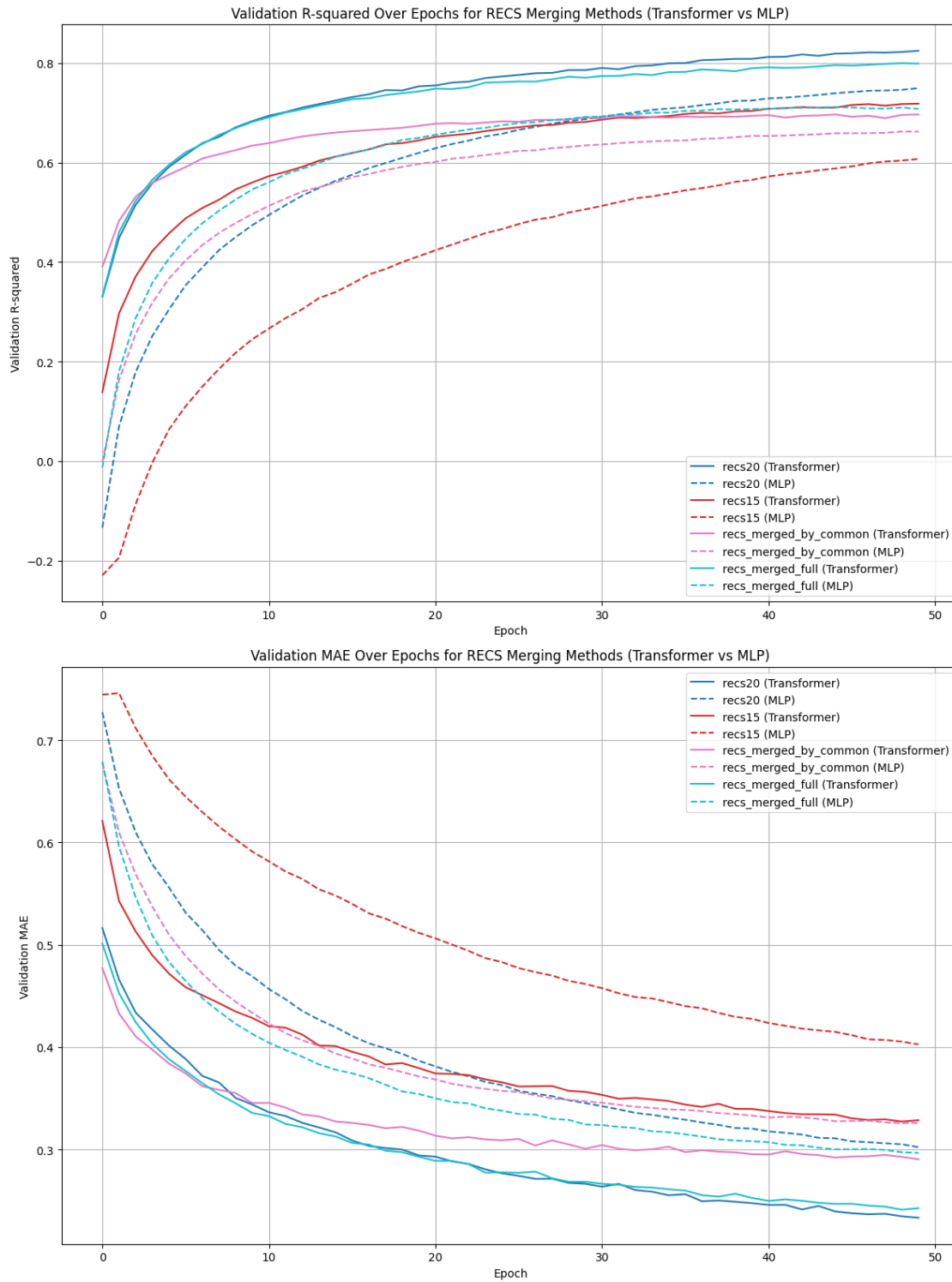


Figure 16: Validation R-squared and MAE per epoch for RECS

The figures present the Validation R-squared and Mean Absolute Error (MAE) across training epochs for Transformer and MLP models applied to the RECS dataset. The MLP models, depicted with dotted lines, consistently show suboptimal performance compared to the Transformer models, which are represented by solid lines. Additionally, models that employ a full merging strategy outperform those using a common merging approach, with the RECS20 dataset achieving the highest performance overall. The MAE graph reinforces this trend, indicating that the Transformer model trained on RECS20 attains the lowest validation MAE. All models exhibit a similar pattern: they rapidly converge within the first 20 epochs and only show slight performance improvements over the subsequent 30 epochs as they stabilize. This behavior suggests that the models quickly learn the essential patterns in the data during the initial training phase, while the later epochs contribute to fine-tuning and marginal enhancements in performance. The superior performance of the Transformer models, especially on the RECS20 dataset, may be attributed to their ability to capture complex relationships in the data more effectively than MLPs.

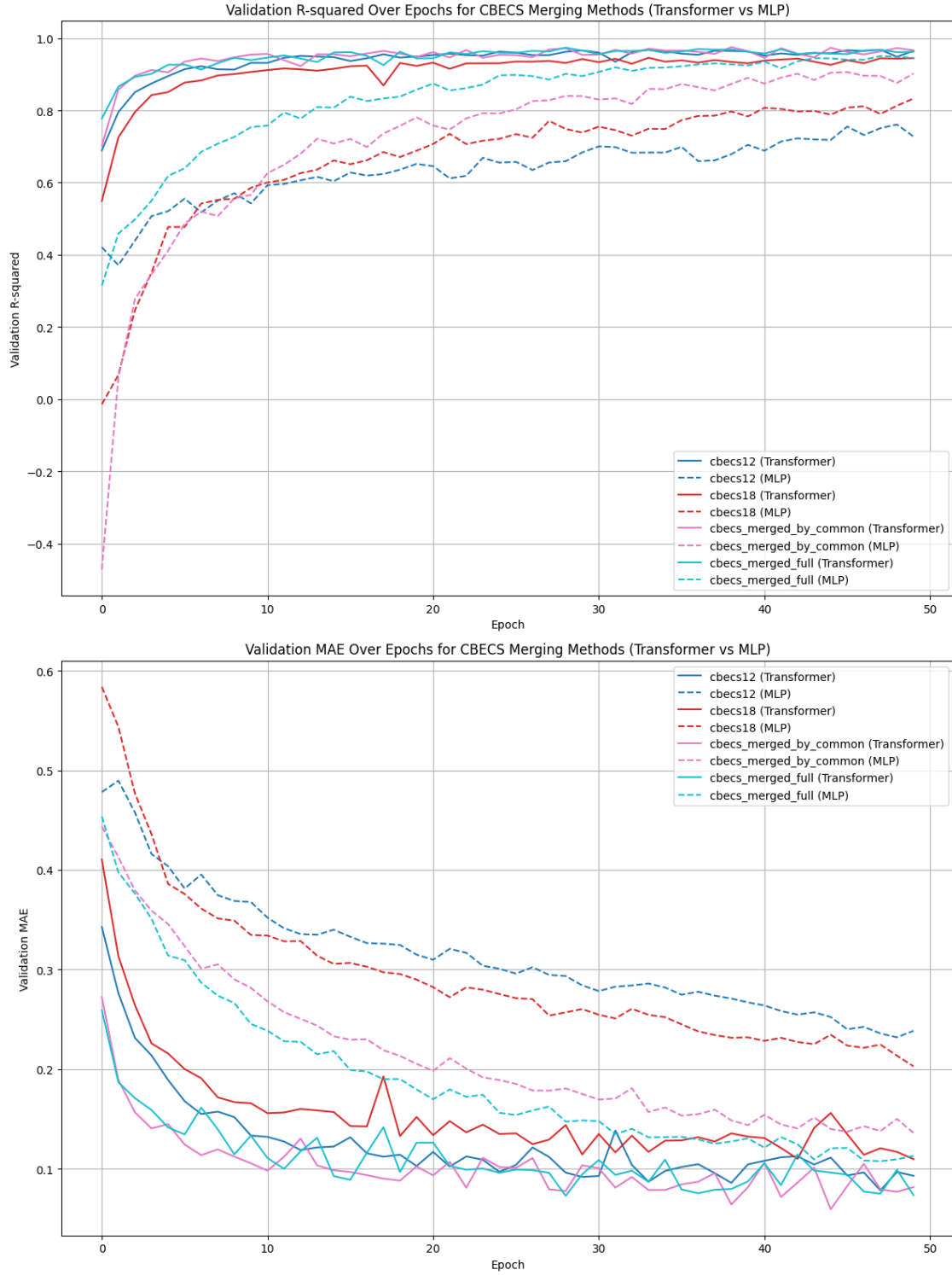


Figure 17: Validation R-squared and MAE per epoch for CBECS

The figures illustrate the Validation R-squared and Mean Absolute Error (MAE) across training epochs for Transformer and MLP models applied to the CBECS dataset. Similar to the RECS models, all Transformer models demonstrate comparable performance levels, with



CBECS18 being the lowest-performing Transformer variant. In contrast, the MLP models exhibit more distinguishable performance differences, with CBECS12 and CBECS18 representing the poorest performers. Furthermore, models that utilize a full merging strategy show a clear advantage, performing nearly on par with the Transformer models. This suggests that the full merging approach enhances the MLP models’ ability to capture complex patterns in the data, thereby narrowing the performance gap between MLPs and Transformers. The consistent performance patterns across both datasets indicate that Transformer models generally maintain a stable and superior performance, while MLP models benefit significantly from effective data merging strategies. This behavior underscores the Transformers’ inherent capability to model intricate relationships within the data, making them a robust choice for such tasks.

Compared to the RECS datasets, the CBECS models exhibit significantly higher oscillations during training, particularly evident in the R-squared values, which occasionally experience steep drops. These fluctuations indicate that the CBECS models may face greater challenges in capturing the underlying data patterns consistently. Several factors could contribute to this behavior, such as increased complexity or variability within the CBECS data, differences in feature distribution, or smaller sample sizes that make the models more susceptible to overfitting. Additionally, the higher oscillations might reflect instability in the training process, potentially requiring more robust tuning of hyperparameters or the use of regularization techniques to achieve smoother convergence. Despite these challenges, the application of full merging strategies in MLP models still provides notable advantages, allowing them to perform more comparably to Transformer models even within the more volatile CBECS dataset. This underscores the importance of effective data integration methods in enhancing model performance, particularly in datasets that present greater inherent variability.

## 4.4 Transformation, and Merging Methods During Training

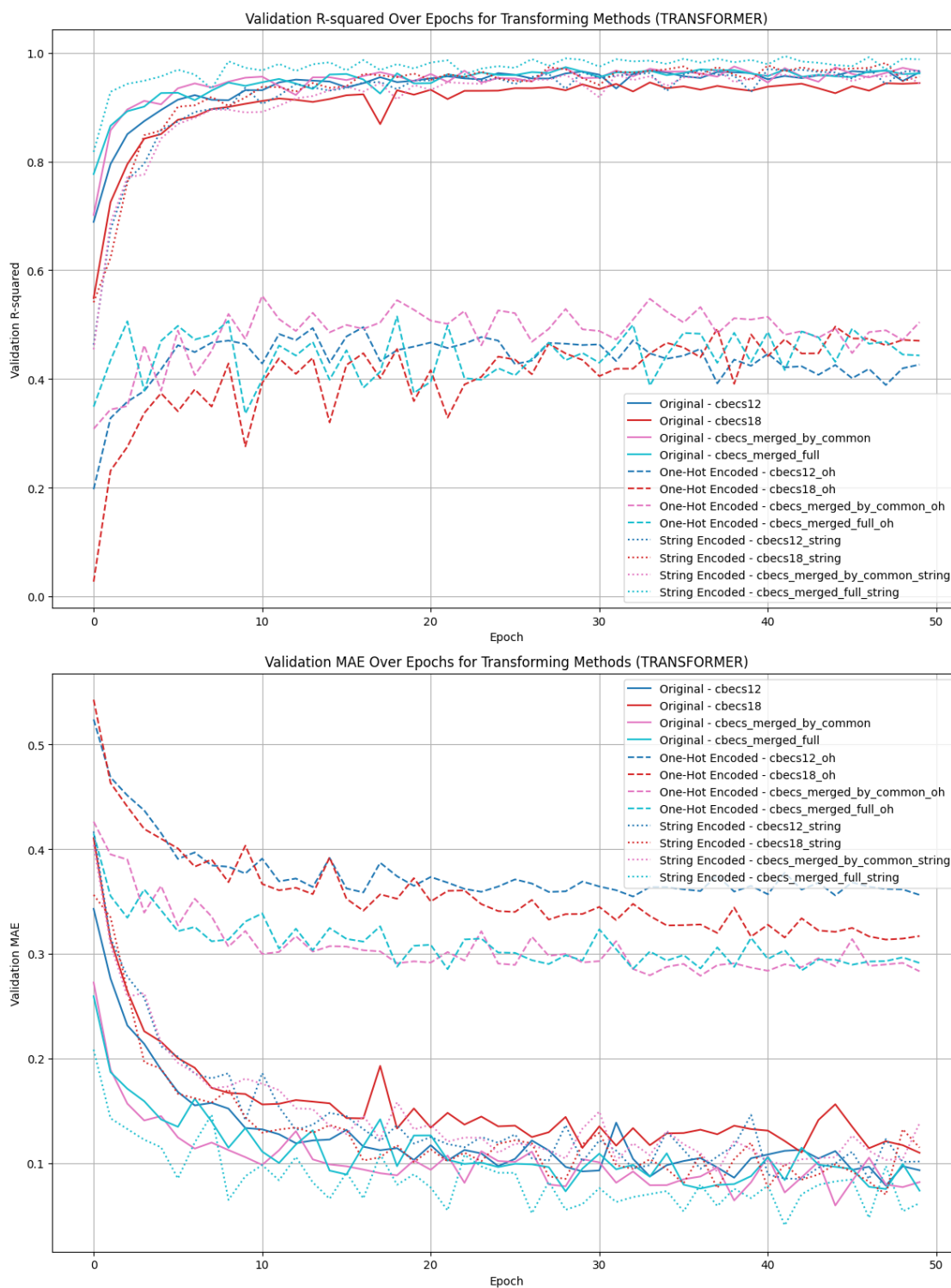


Figure 18: Validation R-squared and MAE per epoch for Transformer models under transformations

The figure presents the validation R-squared and Mean Absolute Error (MAE) per epoch for Transformer models applied to the CBECS dataset, utilizing three distinct types of data transformations. The solid lines represent the base models without any transformation, the dotted lines correspond to models with one-hot encoded features, and the finely dotted lines indicate models employing DistilBERT-encoded string representations. The results demonstrate a slight improvement in performance for the DistilBERT-encoded transformations compared to the original base models, particularly when the data is merged in full, which achieves the highest performance among the tested configurations. This enhancement may be attributed to DistilBERT’s ability to capture more nuanced semantic information from the input data, thereby providing richer feature representations for the Transformer models. All models seem to converge in the first 10 epoch with minimal improvement in the last 40 epochs.

In contrast, the one-hot encoded models consistently perform significantly worse than both the untransformed and DistilBERT-transformed models. This degradation in performance could be due to the sparsity and high dimensionality introduced by one-hot encoding, which may lead to inefficiencies in learning and hinder the model’s ability to generalize effectively.

The MAE trends mirror those observed in the R-squared metrics, reinforcing the correlations between these evaluation measures. Notably, the MAE curves exhibit greater oscillations compared to the smoother R-squared trajectories. This increased variability in MAE may result from the MAE’s sensitivity to outliers and its absolute error nature, which can cause more pronounced fluctuations during training. Overall, the findings highlight the advantages of using advanced encoding techniques like DistilBERT for enhancing Transformer model performance, while also underscoring the limitations of simpler encoding methods such as one-hot encoding in handling complex datasets like CBECS.

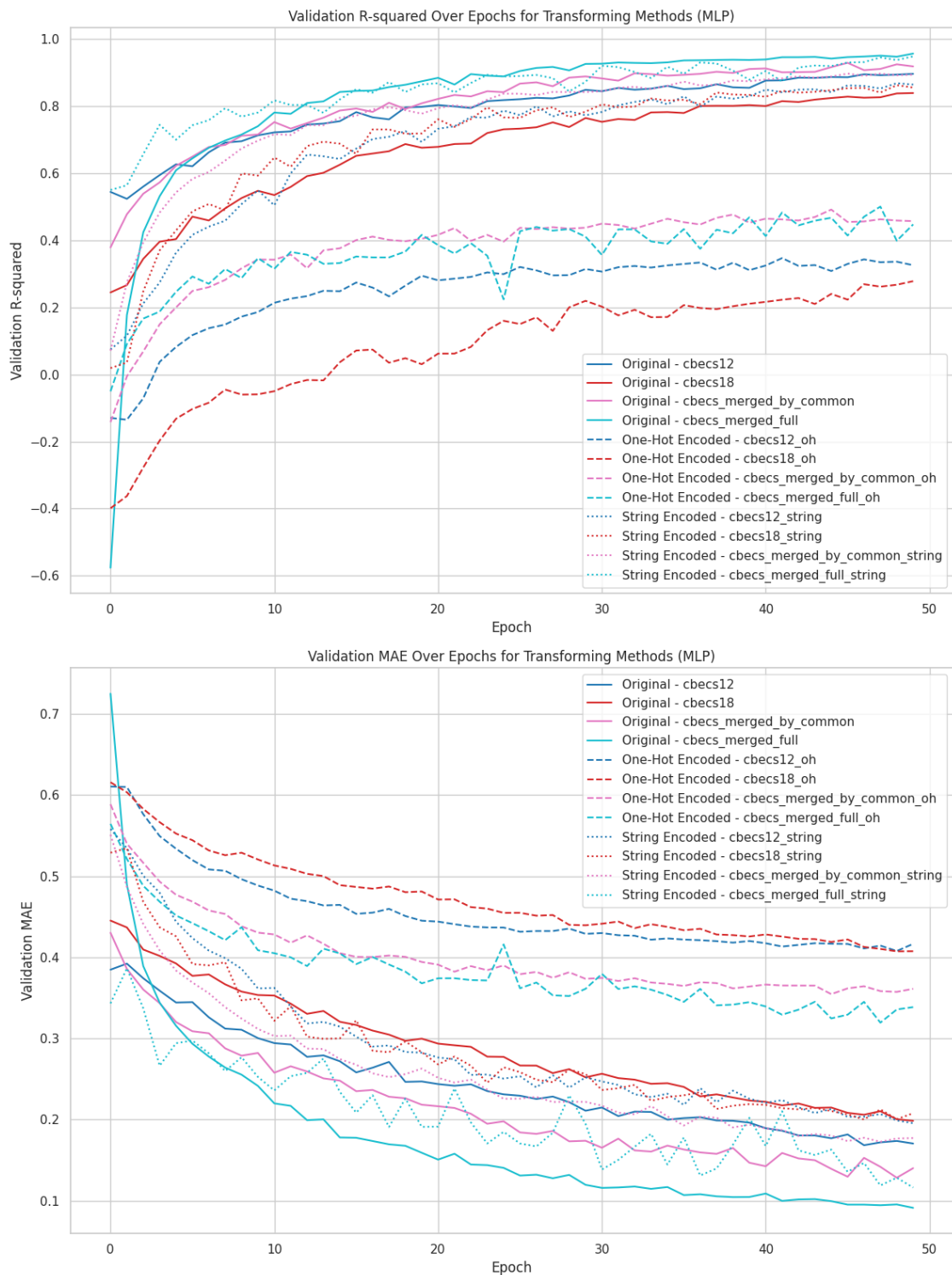


Figure 19: Validation R-squared and MAE per epoch for MLP models under transformations

The figure presents the validation R-squared and Mean Absolute Error (MAE) per epoch for MLP models applied to the CBECS dataset, utilizing three distinct types of data transformations. The observed patterns closely mirror those of the Transformer models; however,

the base MLP models exhibit higher variations. Specifically, the models without transformation and those using DistilBERT transformations perform worse, with one-hot encoded models showing very similar performance, thereby diminishing the differences between them. The base models experience greater variability, with the fully merged base model performing the best, while the base CBECS12 model ranks as the worst among all non-one-hot encoded models.

This might be because MLP architectures are inherently more sensitive to the nature of the input data transformations. One-hot encoding, despite increasing dimensionality, provides a sparse and straightforward representation that MLPs can effectively process without introducing excessive complexity. In contrast, DistilBERT transformations generate more intricate feature representations that may not align as well with the MLP’s capacity to capture complex patterns, leading to suboptimal performance and increased variability. Additionally, the higher variations observed in the base models suggest that without robust feature engineering, MLPs may struggle to generalize effectively on the CBECS dataset, highlighting the importance of appropriate data preprocessing techniques in enhancing model stability and performance.

## 4.5 Model training results

Refer to appendix [10.1](#) for all training plots

Across all 32 training figures, a consistent pattern emerges when comparing the performance of Transformer models and Multi-Layer Perceptrons (MLPs). Specifically, the validation performance of both models is generally similar, with Transformers often having a slight edge. This indicates that, in terms of generalizing to unseen data, both architectures perform comparably, though Transformers may have a marginal advantage in capturing the necessary patterns.

However, a distinct divergence is observed in their training performances. Transformers consistently outperform MLPs during training, achieving higher performance and better fit to the training data. In contrast, MLPs lag significantly behind, suggesting that they are not capturing the underlying complexities of the training dataset as effectively. This disparity points to a couple of critical issues: the Transformer models may be overfitting the training data, learning not only the genuine patterns but also the noise, which can limit their generalization despite their strong performance on validation data. On the other hand, MLPs appear to be underfitting, indicating that they might be too simplistic or lack sufficient capacity to model the intricate relationships within the data.

Several factors could contribute to this phenomenon. Transformers, with their advanced architecture and attention mechanisms, have a higher capacity to learn complex patterns, which can lead to overfitting if not properly regularized. In contrast, MLPs might not have the necessary depth or complexity to capture these patterns, resulting in underfitting. Additionally, differences in hyperparameter settings, such as learning rates, regularization techniques, or the amount of training data, could further influence these outcomes.

Understanding and addressing these factors is crucial for balancing model complexity and generalization to achieve optimal performance across different architectures.

## 4.6 Model training time and inference time

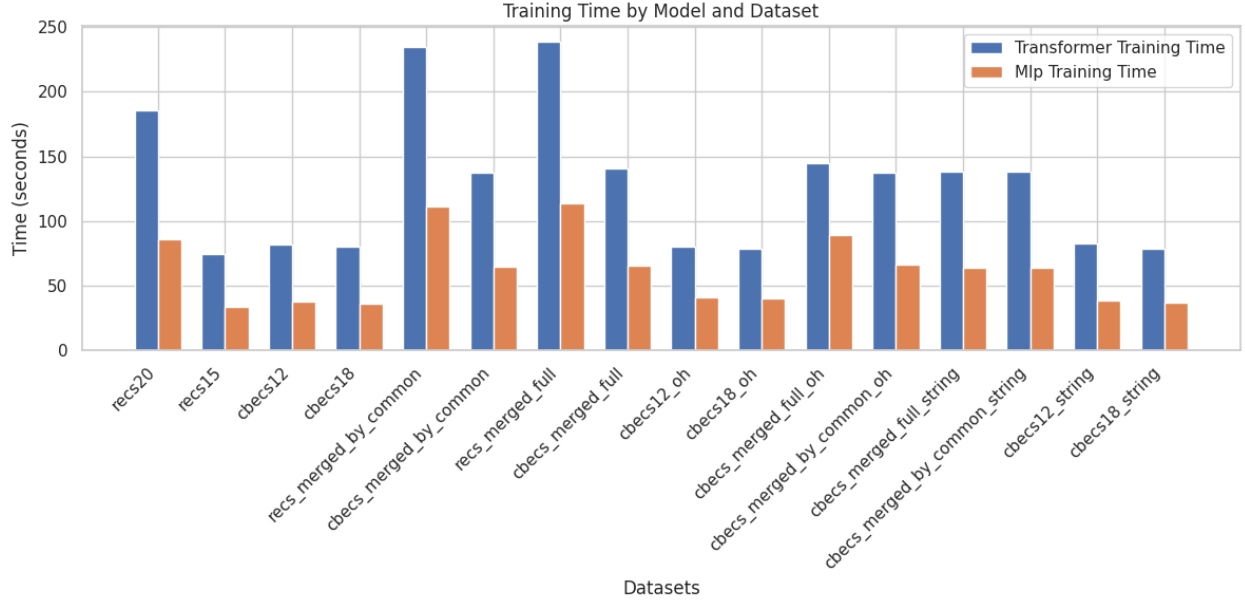


Figure 20: Model training time

The figure above illustrates the training times for various deep learning models. The transformer consistently requires more training time, likely due to its greater complexity and deeper architecture. Models that merge data sources exhibit higher training times overall, with the full merged model for the RECS dataset having the longest training duration. In general, models trained on the RECS dataset tend to take longer than those trained on CBECS, potentially because of differences in data characteristics or the complexity of the RECS dataset. This

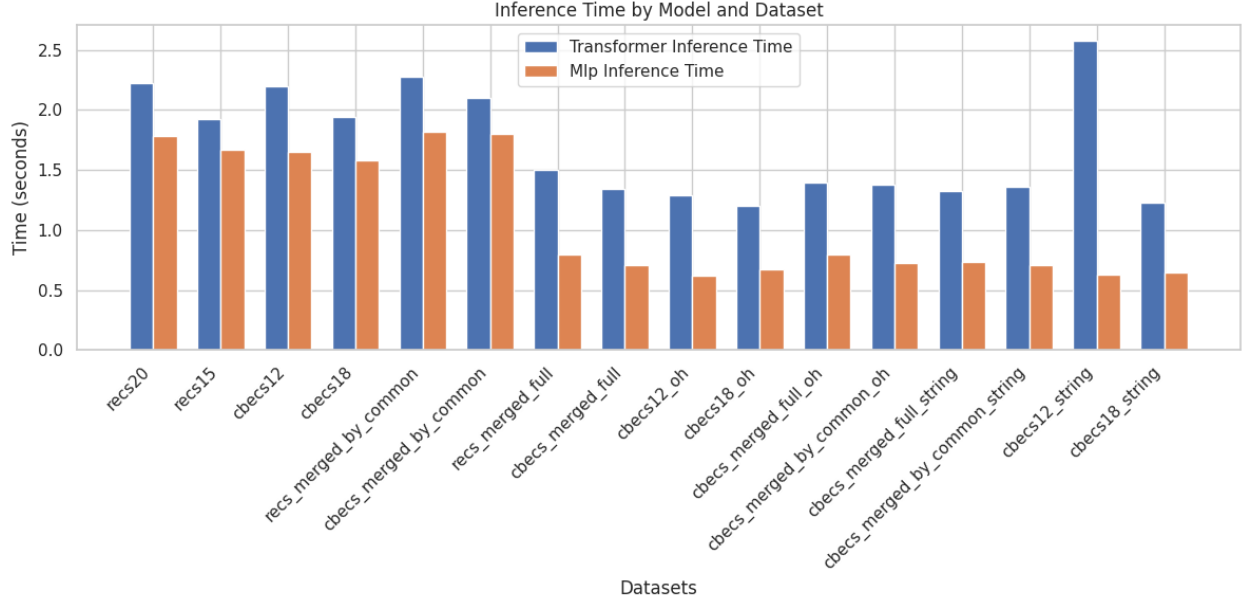


Figure 21: Model inference time

The figure shows the inference time (time taken by the model to make a prediction) across various deep learning models. Transformer models generally have longer inference times compared to MLPs, likely due to their increased complexity. The base models show relatively higher inference times, but notably, the DistilBERT transformer model for the CBECS12 dataset experiences an unexpected spike in inference time, making it the highest among all models analyzed. This surge could be due to the specific characteristics of the dataset or additional overhead in the model architecture.

## 5 Discussion

In our study, several limitations impacted the model’s reliability and generalization. Due to time constraints, we could not perform thorough manual integrity checks, specifically in verifying the validity of certain features and ensuring they were relevant before energy consumption occurred. Additionally, we faced challenges in detecting potential data leakage during the merging of datasets, which could affect the performance of predictions. Both of these issues highlight the need for more comprehensive data validation processes that were not feasible within the time and resource constraints of this project.

### 5.1 Limitations and recommendations

Due to hardware limitations, our ability to conduct extensive hyperparameter tuning was significantly constrained. Specifically, we were unable to explore a wide range of structural parameters such as the number of blocks, various hidden size configurations, and other nuanced hyperparameters like dropout rates and the number of attention heads. Hyperparameter tuning is crucial for optimizing model performance, as it allows for the fine-tuning

of the model architecture to best fit the data. Without adequate computational resources, our exploration was limited to a narrower set of configurations, potentially preventing us from identifying the most effective combinations that could enhance model performance and generalization. This constraint underscores the importance of having access to more robust hardware or leveraging cloud-based solutions to facilitate comprehensive model optimization in future studies.

Another significant limitation of our study was the inability to perform higher-fold cross validations on each model. Cross validation is a vital technique for assessing the robustness and generalizability of machine learning models by partitioning the data into multiple subsets and iteratively training and testing the model on different combinations. Higher-fold cross validations, such as 10-fold or even 20-fold, provide a more reliable estimate of model performance by reducing variance and mitigating the impact of data anomalies. However, due to time and computational constraints, we were restricted to fewer folds, which may have led to less precise performance metrics and a higher risk of overfitting or underfitting. Future research would benefit from incorporating higher-fold cross validations to ensure more dependable and comprehensive evaluations of model performance across diverse datasets.

**Lack of Advanced Deep Learning Expertise for Custom Neural Network Design** Our project was also hindered by the lack of advanced expertise in deep learning, which limited our capacity to design and implement specialized neural networks with custom feedforward and backward architectures. Developing bespoke neural network architectures often requires deep knowledge of intricate model components and optimization techniques to tailor the networks to specific tasks effectively. Without this expertise, we were constrained to using standard architectures, which may not fully capture the unique patterns and complexities inherent in our data. This limitation potentially restricted the performance ceiling of our models, as customized architectures could offer enhanced flexibility and efficiency in learning from the data. Investing in advanced training or collaborating with deep learning experts in future projects could enable the creation of more sophisticated models, thereby improving overall performance and applicability.

Another challenge we faced was the absence of advanced expertise in civil engineering and building systems, which impeded our ability to perform heuristic feature selection. Instead of leveraging domain-specific knowledge to manually select and prioritize features based on their relevance and impact, we had to rely on the model's inherent capability to automatically weight each feature. Heuristic feature selection can significantly enhance model performance by focusing on the most pertinent variables and reducing noise from irrelevant ones. Without this targeted approach, the model may have been exposed to redundant or less informative features, potentially diminishing its predictive performance and interpretability. Incorporating expertise from civil engineering in future work would allow for more informed feature selection, thereby improving the model's efficiency and effectiveness in addressing domain-specific challenges.

Time constraints also prevented us from manually merging datasets, specifically the Residential Energy Consumption Survey (RECS) with the Commercial Building Energy Con-



sumption Survey (CBECS). Merging these datasets could have provided a more comprehensive and diverse data pool, enhancing the robustness and generalizability of our models. Manual merging involves aligning data structures, resolving inconsistencies, and ensuring the integrity of combined datasets, which is a time-consuming process. Due to limited time, we had to proceed without this potentially valuable integration, which may have restricted the breadth of our analysis and the applicability of our findings across different building types and energy consumption patterns. Future studies should allocate sufficient time and resources to undertake meticulous dataset merging, thereby enriching the data landscape and enabling more thorough and versatile model evaluations.

Finally, our evaluation was constrained by dataset insufficiencies, as we relied solely on the RECS and CBECS datasets, both of which are based on data from New York. This geographical limitation may have introduced biases specific to the region’s building practices, climate conditions, and energy usage patterns, potentially limiting the generalizability of our models to other regions with different characteristics. Incorporating additional datasets from diverse geographical locations and varied survey sources would provide a more comprehensive foundation for model training and evaluation, enhancing their applicability and robustness across different contexts. Expanding the dataset repertoire in future research would help mitigate regional biases and ensure that the models developed are more universally applicable, thereby increasing their utility and impact in addressing energy consumption challenges on a broader scale.

The limited time available also prevented us from conducting thorough manual integrity checks, particularly in verifying whether each feature is valid and should exist before energy consumption occurs, ensuring that the predictions are meaningful. Additionally, we were unable to fully assess whether data leakage issues arose during the merging of datasets. Addressing these concerns would require more detailed and specific data to validate the results, which we were unable to obtain within the scope of this project. These checks are crucial to ensure the reliability and performance of the model’s predictions and to avoid any unintended biases or errors from improperly handled data.

## 5.2 Future works

Future work should prioritize the implementation of more rigorous data integrity checks to ensure that all features are valid and pertinent to energy consumption predictions. This involves a deeper analysis of each feature to confirm its relevance and timing in relation to the energy usage being modeled. In cases where the features may not directly precede energy consumption, additional filtering or feature engineering techniques may be required to prevent irrelevant or misleading information from impacting the model’s performance. Furthermore, the development of automated tools for detecting potential data leakage during dataset merging would help safeguard the model from biases introduced by unintended correlations between training and test data. This could be achieved by applying cross-validation methods specifically tailored to evaluate data consistency across merged datasets.

Another avenue for future work is acquiring additional data to validate the results more

effectively and address any underlying issues. Access to more diverse datasets, especially those with region-specific or temporal variations, would enable a more comprehensive assessment of the model’s robustness and generalizability. With more time and resources, the research could also explore more advanced techniques, such as domain-specific feature selection and model-specific adjustments, to further enhance the prediction performance. This would allow for a more thorough exploration of how different features interact with energy consumption and minimize the risk of overfitting or data leakage during training. By addressing these aspects, future studies can ensure more reliable and meaningful outcomes.

The anomalous R-squared and MAE values observed for XGBoost, especially on the CBECS12 subset, warrant a deeper investigation into potential causes. Future work should explore data preprocessing steps, feature engineering inconsistencies, and model hyperparameter settings that may lead to overfitting or underfitting. Additionally, examining the stability of model performance across different data splits can help identify underlying issues related to data variability or model robustness. Overall, while Transformer-based deep learning models clearly outperform other models across most datasets, the identified anomalies underscore the necessity for thorough model validation and diagnostic analysis to ensure reliability and consistency in real-world applications.

Future work in models should focus on addressing the observed overfitting in Transformer models and the underfitting in Multi-Layer Perceptrons (MLPs) to enhance overall performance and generalization. One promising direction is to implement advanced regularization techniques for Transformers, such as dropout, weight decay, or data augmentation, to mitigate overfitting while maintaining their capacity to capture complex patterns. Additionally, exploring architectural modifications, such as reducing the number of layers or attention heads, could help balance model complexity with generalization capabilities. For MLPs, increasing the network depth or incorporating more sophisticated activation functions and connectivity patterns might alleviate underfitting by enabling the models to learn more intricate relationships within the data. Another avenue is to experiment with hybrid models that combine the strengths of Transformers and MLPs, potentially leveraging the attention mechanisms of Transformers alongside the simplicity and efficiency of MLPs. Furthermore, conducting comprehensive hyperparameter optimization for both model types could identify optimal settings that enhance performance and reduce discrepancies between training and validation results. Finally, expanding the scope of experiments to include diverse datasets and tasks would provide deeper insights into the generalizability of these models and inform the development of more robust and versatile neural architectures. By systematically exploring these strategies, future research can achieve a more balanced performance between Transformers and MLPs, leading to models that are both powerful and resilient in various applications.

Future work on data acquisition should adopt a more flexible approach, as our findings suggest that altering features can be effective as our data merging in full methods sees a performance improvement. The experiments with DistilBERT demonstrate that incorporating specialized notations or handling unique data structures such as text can still yield valuable insights. By highlighting the potential of survey data in building energy predictions, we

advocate for its expanded collection to complement sensor or simulation data, which are often limited in availability within the existing building energy prediction literature. Survey data can serve as a valuable supplement to address the current data scarcity and enhance the robustness of energy prediction models. Ultimately, we hope our methods for handling high-dimensional and noisy survey data can provide a foundation for future researchers, equipping them with tools to overcome common challenges in survey-based studies. This flexibility in data handling will help researchers better adapt to the complexities of survey data, fostering more robust and innovative research in building energy prediction.

## 6 Conclusion

Our research question centered on identifying the best methods for leveraging survey data in building energy prediction. The study examined the performance of Transformer, Multi-Layer Perceptron (MLP), XGBoost, and Support Vector Machine (SVM) models on various RECS and CBECS datasets using R-squared and Mean Absolute Error (MAE) metrics. Across the board, Transformer models consistently outperformed the others, achieving the highest R-squared values and demonstrating superior predictive performance. Their ability to capture complex feature interactions through attention mechanisms made them particularly effective on high-dimensional datasets. MLPs followed closely behind the Transformers, performing robustly in most cases but lacking the same level of sophistication in handling complex data patterns. XGBoost and SVM, on the other hand, struggled significantly, especially with datasets like RECS15, where high dimensionality and the absence of extensive feature selection led to poor generalization and low R-squared values.

The study also explored how different feature transformations impacted model performance. The use of DistilBERT embeddings proved to be highly beneficial for the Transformer models, as it provided a compact, semantically rich representation of features, resulting in a substantial performance boost. One-hot encoding, while useful for traditional machine learning models, posed challenges for high-dimensional data, particularly for XGBoost and MLPs, which tended to overfit or underperform when faced with too many irrelevant features. The raw feature space in the base datasets favored Transformers due to their ability to manage intricate feature interactions, while MLPs struggled without the benefit of advanced feature engineering.

Dataset merging strategies also played a crucial role in model performance. When the datasets collected from different times and having uncommon features were merged on common features, performance generally improved, especially for Transformers and MLPs, as the increased dataset size allowed for more robust learning. Full merging, where all feature columns were combined, led to peak performance for Transformers, which could efficiently handle the large-scale data and prioritize relevant features through their self-attention mechanisms. While MLPs also saw improvement in merged datasets, they did not capitalize on the complexity of the data as effectively as Transformers. Traditional models like XGBoost and SVM continued to struggle, showing limited ability to generalize in these high-dimensional settings.

Training dynamics revealed that Transformers consistently outperformed MLPs during training, quickly converging to a higher performance. This efficiency, however, raised concerns about potential overfitting, particularly in complex datasets. In contrast, MLPs exhibited slower convergence and signs of underfitting, suggesting they lacked the capacity to fully capture the intricate relationships within the data. This indicates that while Transformers may require regularization techniques to prevent overfitting, MLPs could benefit from architectural adjustments or improved hyperparameter tuning to enhance their performance.

Overall, the study recommends prioritizing Transformers and MLPs for deployment in prac-

tical applications, given their demonstrated effectiveness across datasets. Advanced feature engineering, such as the use of DistilBERT embeddings, should be employed to maximize model performance. Additionally, addressing model limitations—such as overfitting in Transformers and underfitting in MLPs—through regularization and architectural refinement is crucial. Incorporating additional evaluation metrics like Root Mean Squared Error (RMSE) can also provide deeper insights into model performance, especially in cases where R-squared and MAE provide conflicting results.

Our research has provided valuable insights into the use of survey data for building energy consumption prediction. Through the comparative analysis of different AI models and feature engineering techniques, we have shown that survey data, despite its inherent challenges, can be effectively utilized when paired with the right preprocessing methods and machine learning architectures. As such, this research paves the way for more extensive use of survey data in future building energy prediction models, offering a viable alternative or complement to direct measurement sensory or simulation data. Moving forward, survey data collectors can gather more extensive datasets with greater confidence in their value and potential. Additionally, they have the flexibility to include unstructured data, such as open-ended responses or detailed annotations, as advancements in NLP models now enable effective processing and analysis of such information. Future research can continue to refine these methods, further integrating survey data into the building energy consumption prediction landscape and exploring its potential to improve model accuracy and applicability across different building types.

## 7 Availability

All the code files and data can be found in [repository](#) with public read access. Aside for all the scripts of training and recording models in the models folder and datasets in the datasets folder, the repository also contains the website for demonstration purpose in the website folder. Follow the README.md file for more details.

## 8 Acknowledgement

All data used in this study was obtained from the U.S. Energy Information Administration, in accordance with their licensing terms. For additional permissions regarding the dataset, please refer to their most recent policies.

## 9 References

- [1] Q. Qiao, A. Yunusa-Kaltungo, and R. E. Edwards, “Towards developing a systematic knowledge trend for building energy consumption prediction,”

- Journal of Building Engineering*, vol. 35, p. 101967, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352710220335993>
- [2] K. Amasyali and N. M. El-Gohary, “A review of data-driven building energy consumption prediction studies,” *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 1192–1205, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1364032117306093>
  - [3] “Energyplus™, version 00,” 9 2017, [Accessed: Oct. 24, 2024]. [Online]. Available: <https://www.osti.gov/servlets/purl/1395882>
  - [4] Y. Wei, X. Zhang, Y. Shi, L. Xia, S. Pan, J. Wu, M. Han, and X. Zhao, “A review of data-driven approaches for prediction and classification of building energy consumption,” *Renewable and Sustainable Energy Reviews*, vol. 82, pp. 1027–1047, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S136403211731362X>
  - [5] Z. Wang and R. S. Srinivasan, “A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models,” *Renewable and Sustainable Energy Reviews*, vol. 75, pp. 796–808, 2017.
  - [6] X. Chen, M. M. Singh, and P. Geyer, “Utilizing domain knowledge: Robust machine learning for building energy performance prediction with small, inconsistent datasets,” *Knowledge-Based Systems*, vol. 294, p. 111774, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S095070512400409X>
  - [7] M. Pipattanasomporn, G. Chitalia, J. Songsiri, C. Aswakul, W. Pora, S. Suwankawin, K. Audomvongseeree, and N. Hoonchareon, “Cu-bems, smart building electricity consumption and indoor environmental sensor datasets,” *Scientific Data*, vol. 7, no. 1, p. 241, 2020. [Online]. Available: <https://doi.org/10.1038/s41597-020-00582-3>
  - [8] P. A. Mathew, L. N. Dunn, M. D. Sohn, A. Mercado, C. Custudio, and T. Walter, “Big-data for building energy performance: Lessons from assembling a very large national database of building energy use,” *Applied Energy*, vol. 140, pp. 85–93, 2015. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0306261914012112>
  - [9] X. Zhang, Y. Sun, D.-c. Gao, W. Zou, J. Fu, and X. Ma, “Similarity-based grouping method for evaluation and optimization of dataset structure in machine-learning based short-term building cooling load prediction without measurable occupancy information,” *Applied Energy*, vol. 327, p. 120144, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0306261922014015>
  - [10] T. Xiao, P. Xu, R. He, and H. Sha, “Status quo and opportunities for building energy prediction in limited data context—overview from a competition,” *Applied Energy*, vol. 305, p. 117829, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0306261921011570>
  - [11] Y. Gong, G. Liu, Y. Xue, R. Li, and L. Meng, “A survey on dataset quality in machine learning,” *Information and Software Technology*, vol. 162, p. 107268, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0950584923001222>

- [12] G. Lee, L. Alzamil, B. Doskenov, and A. Termehchy, *A Survey on Data Cleaning Methods for Improved Machine Learning Model Performance*, 2021.
- [13] A. Paullada, I. D. Raji, E. M. Bender, E. Denton, and A. Hanna, “Data and its (dis)contents: A survey of dataset development and use in machine learning research,” *Patterns*, vol. 2, no. 11, 2021, doi: 10.1016/j.patter.2021.100336. [Online]. Available: <https://doi.org/10.1016/j.patter.2021.100336>
- [14] A. Mumuni and F. Mumuni, “Data augmentation: A comprehensive survey of modern approaches,” *Array*, vol. 16, p. 100258, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2590005622000911>
- [15] H. Fang, H. Tan, R. Kosonen, X. Yuan, K. Jiang, and R. Ding, “Study of the data augmentation approach for building energy prediction beyond historical scenarios,” 2023.
- [16] Y. Deng, R. Liang, D. Wang, A. Li, and F. Xiao, “Decomposition-based data augmentation for time-series building load data,” p. 51–60, 2023. [Online]. Available: <https://doi.org/10.1145/3600100.3623727>
- [17] C. Robinson, B. Dilkina, J. Hubbs, W. Zhang, S. Guhathakurta, M. A. Brown, and R. M. Pendyala, “Machine learning approaches for estimating commercial building energy consumption,” *Applied energy*, vol. 208, pp. 889–904, 2017.
- [18] H. Deng, D. Fannon, and M. J. Eckelman, “Predictive modeling for us commercial building energy use: A comparison of existing statistical and machine learning algorithms using cbecs microdata,” *Energy and Buildings*, vol. 163, pp. 34–43, 2018.
- [19] D. Rai, “Predicting energy consumption in commercial buildings using its property features and machine learning algorithms,” Ph.D. dissertation, Dublin, National College of Ireland, 2019.
- [20] J. W. Burnett and L. L. Kiesling, “How do machines predict energy use? comparing machine learning approaches for modeling household energy demand in the united states,” *Energy Research & Social Science*, vol. 91, p. 102715, 2022.
- [21] U. E. I. Administration, “2020 residential energy consumption survey (recs) microdata,” <https://www.eia.gov/consumption/residential/data/2020/index.php?view=microdata>, 2020, [Accessed: Oct. 23, 2024].
- [22] —, “2015 residential energy consumption survey (recs) microdata,” <https://www.eia.gov/consumption/residential/data/2015/index.php?view=microdata>, 2015, [Accessed: Oct. 23, 2024].
- [23] —, “2018 commercial buildings energy consumption survey (cbecs) microdata,” <https://www.eia.gov/consumption/commercial/data/2018/index.php?view=microdata>, 2018, [Accessed: Oct. 23, 2024].

- [24] —, “2012 commercial buildings energy consumption survey (cbeccs) microdata,” <https://www.eia.gov/consumption/commercial/data/2012/index.php?view=microdata>, 2012, [Accessed: Oct. 23, 2024].
- [25] A. F. Adoma, N.-M. Henry, and W. Chen, “Comparative analyses of bert, roberta, distilbert, and xlnet for text-based emotion recognition,” in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE, 2020, pp. 117–121.
- [26] S. K. Nayak and A. C. Ojha, “Data leakage detection and prevention: Review and research directions,” *Machine Learning and Information Processing: Proceedings of ICM-LIP 2019*, pp. 203–212, 2020.
- [27] H. Liu, H. Sun, H. Mo, and J. Liu, “Analysis and modeling of air conditioner usage behavior in residential buildings using monitoring data during hot and humid season,” *Energy and Buildings*, vol. 250, p. 111297, 2021.
- [28] A. Vaswani, “Attention is all you need,” *Advances in Neural Information Processing Systems*, 2017.
- [29] V. Thakkar, S. Tewary, and C. Chakraborty, “Batch normalization in convolutional neural networks—a comparative study with cifar-10 data,” in *2018 fifth international conference on emerging applications of information technology (EAIT)*. IEEE, 2018, pp. 1–5.
- [30] P. Baldi and P. J. Sadowski, “Understanding dropout,” *Advances in neural information processing systems*, vol. 26, 2013.
- [31] D. Chicco, M. J. Warrens, and G. Jurman, “The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation,” *Peerj computer science*, vol. 7, p. e623, 2021.
- [32] Z. Zhang, “Improved adam optimizer for deep neural networks,” in *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*. Ieee, 2018, pp. 1–2.
- [33] V. Binson, M. Subramoniam, Y. Sunny, and L. Mathew, “Prediction of pulmonary diseases with electronic nose using svm and xgboost,” *IEEE Sensors Journal*, vol. 21, no. 18, pp. 20 886–20 895, 2021.



## 10 Appendix

### 10.1 Training results

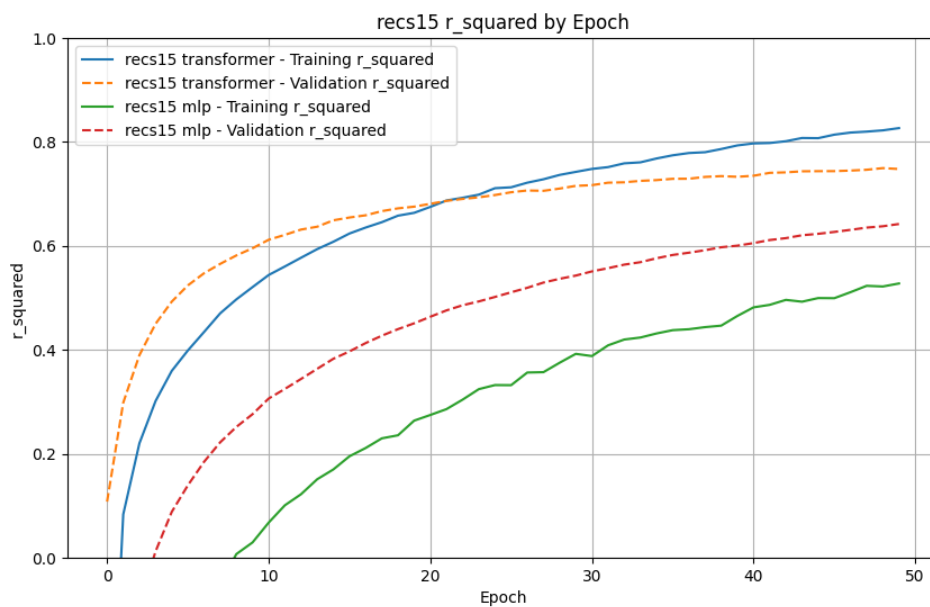


Figure 22: RECS15 R2

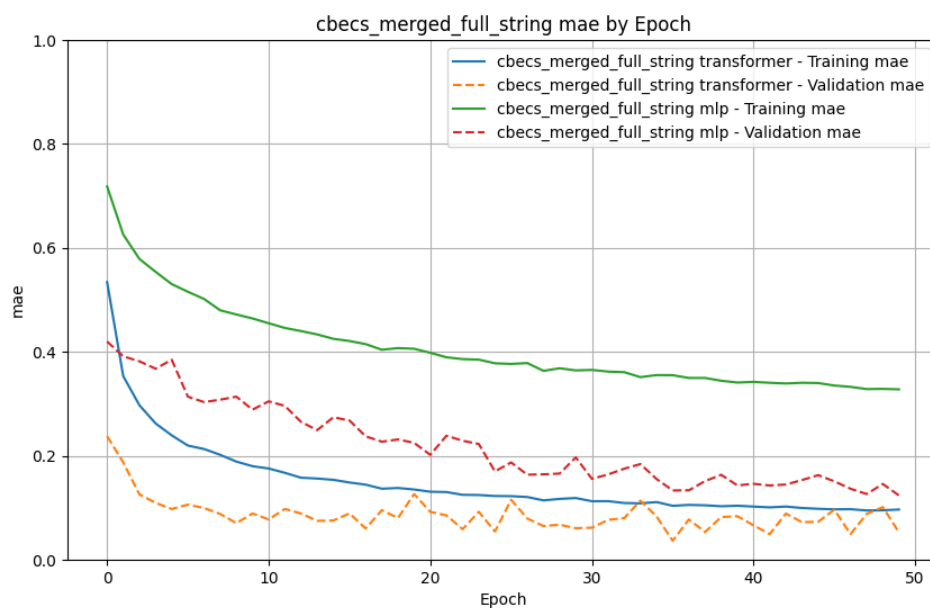


Figure 53: CBECS merged in full DistilBERT MAE

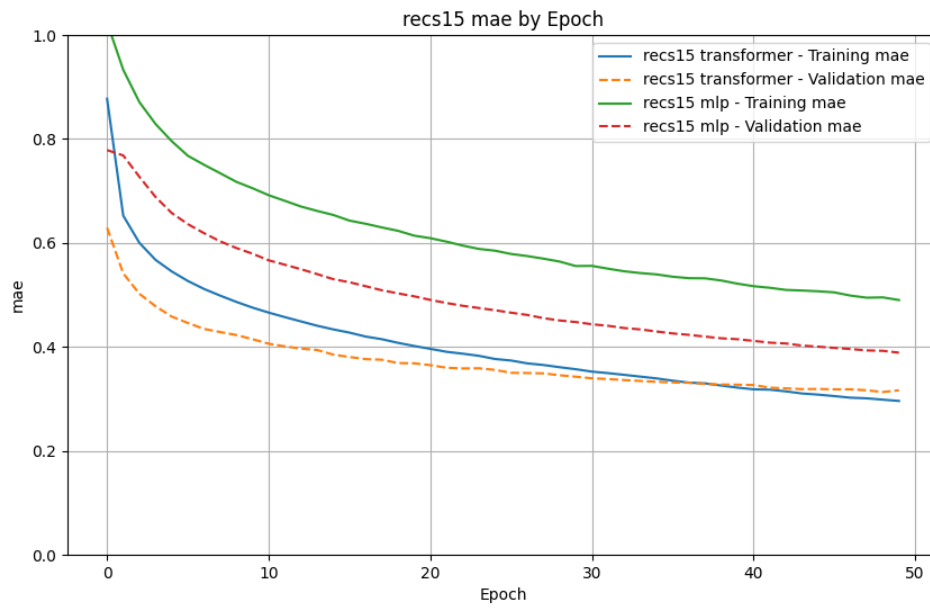


Figure 23: RECS15 MAE

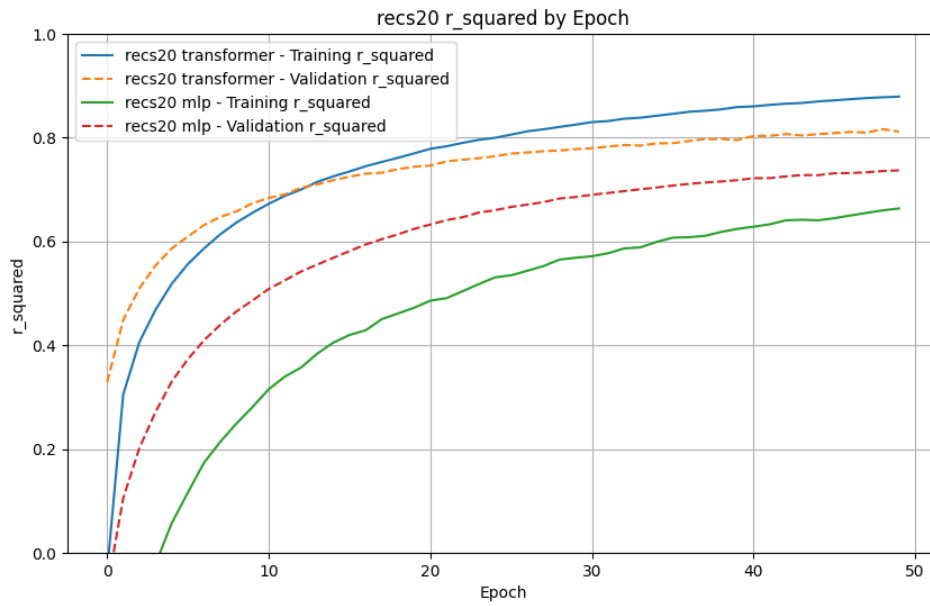


Figure 24: RECS20 R2

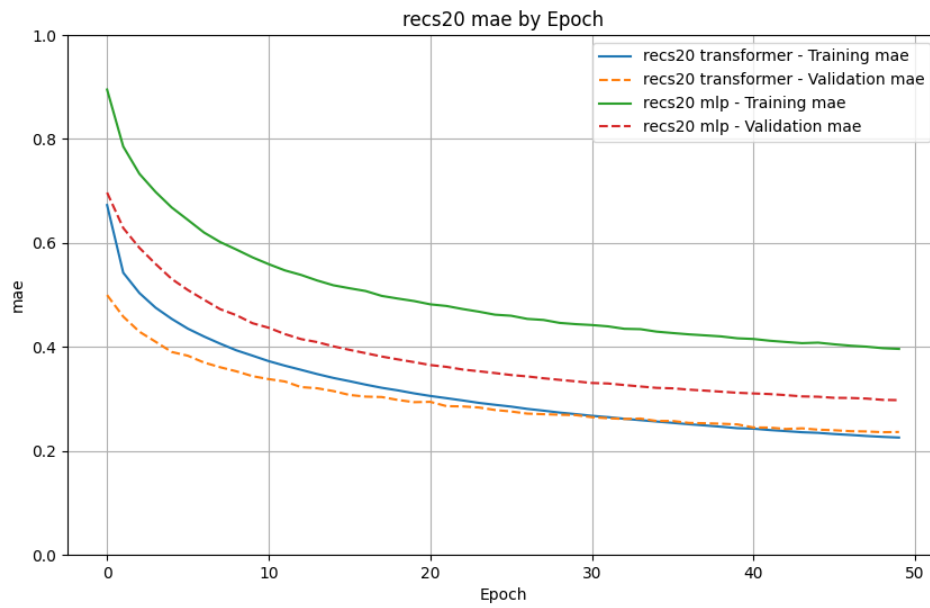


Figure 25: RECS20 MAE

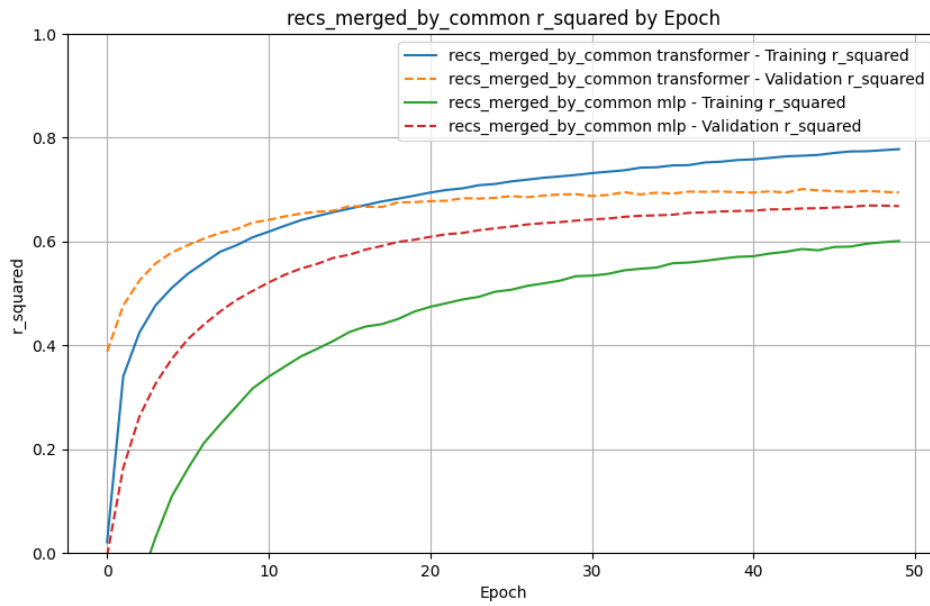


Figure 26: RECS merged by common R2

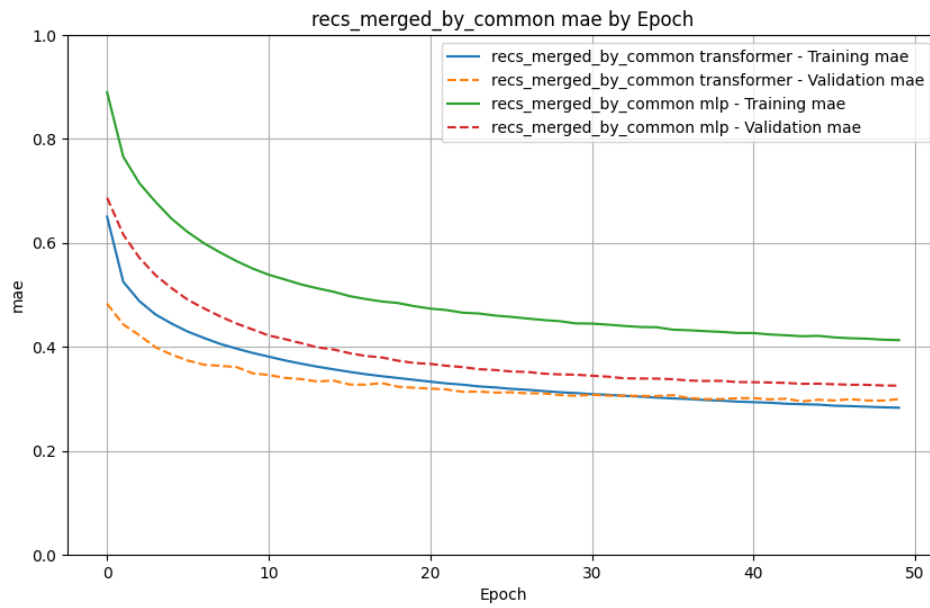


Figure 27: RECS merged by common MAE

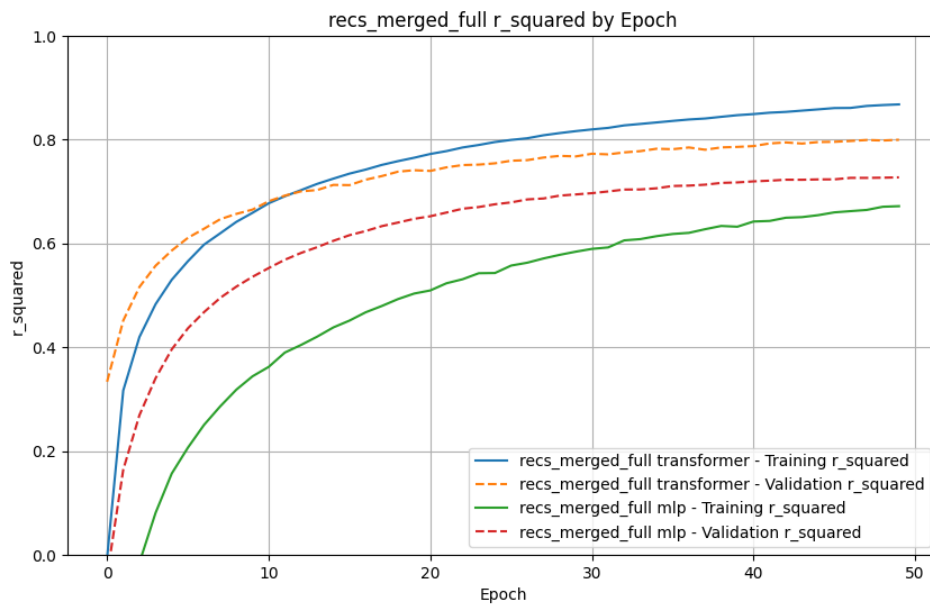


Figure 28: RECS merged in full R2

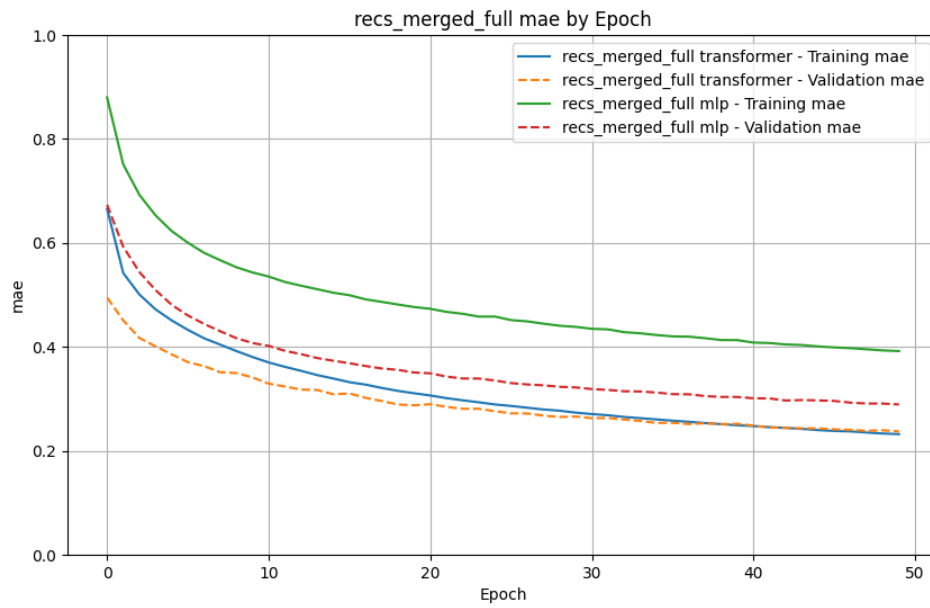


Figure 29: RECS merged in full MAE

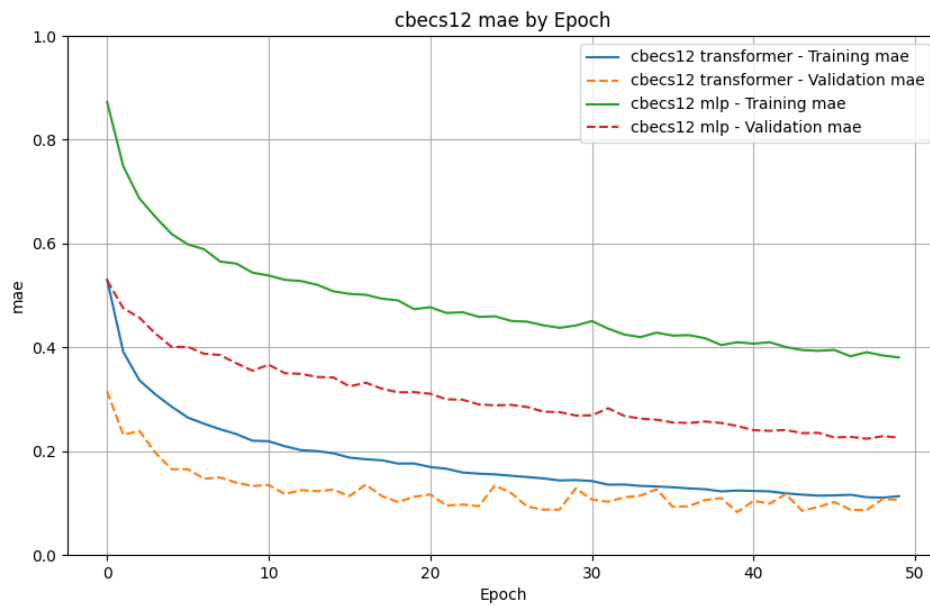


Figure 30: CBECS12 R2

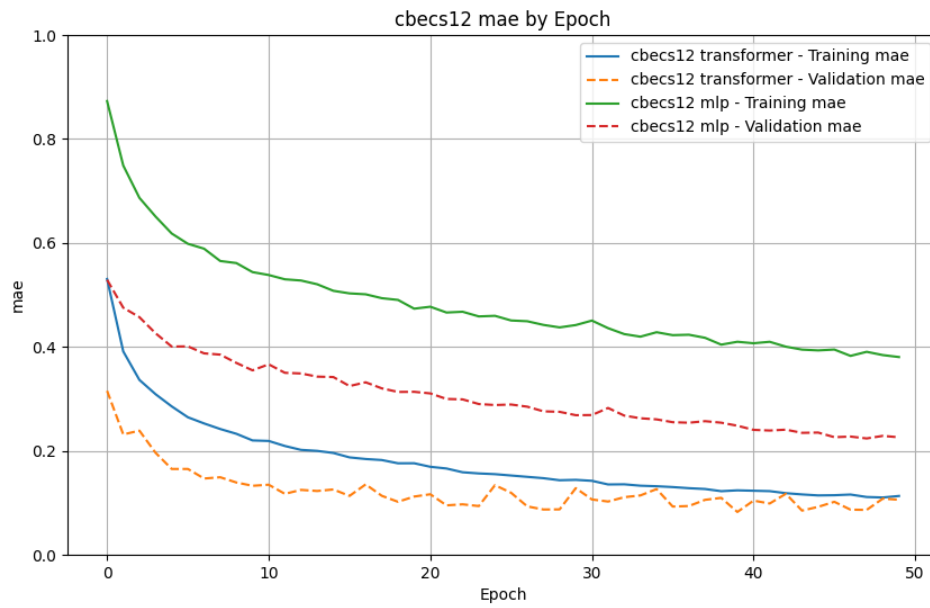


Figure 31: CBECS12 MAE

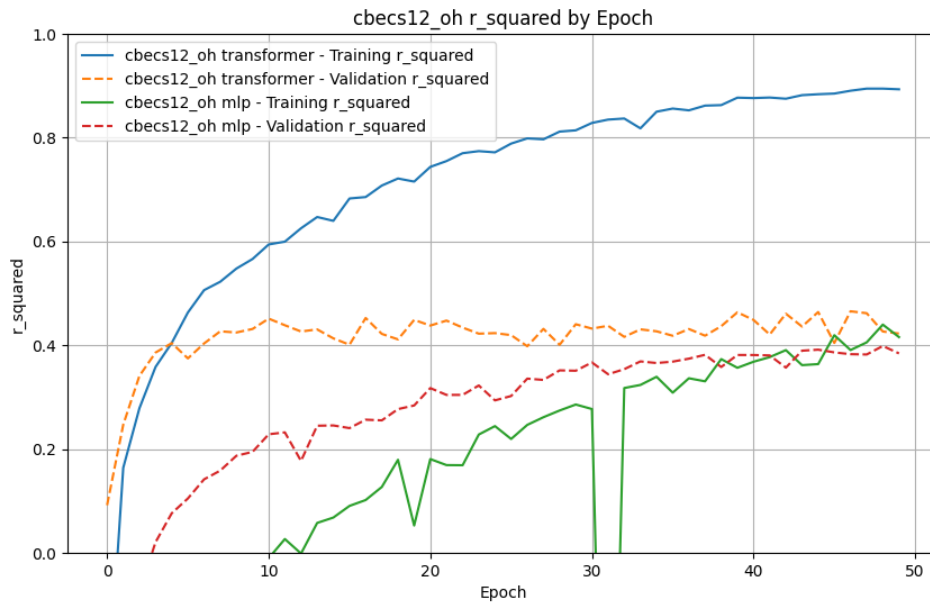


Figure 32: CBECS12 One Hot  $r^2$

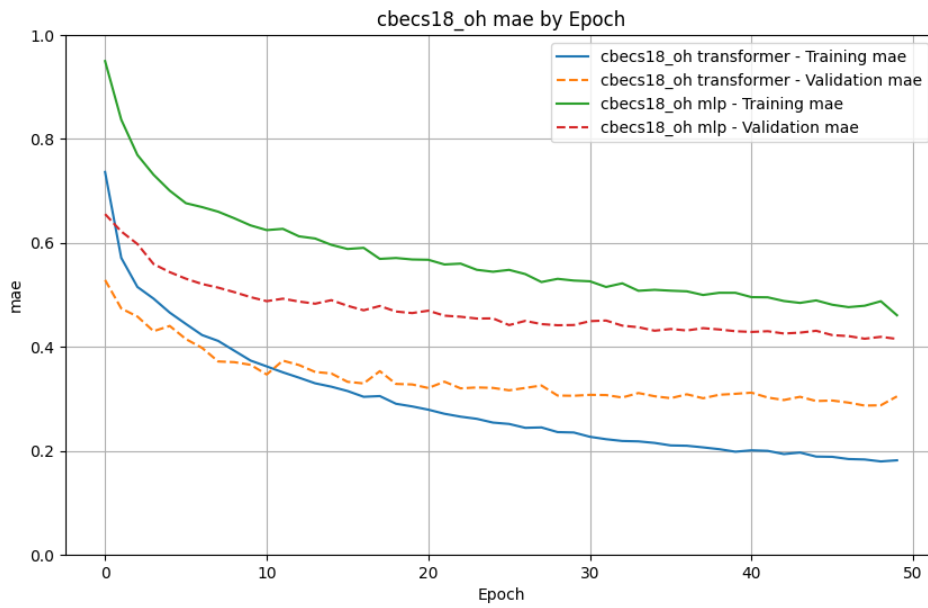


Figure 33: CBECS12 One Hot MAE

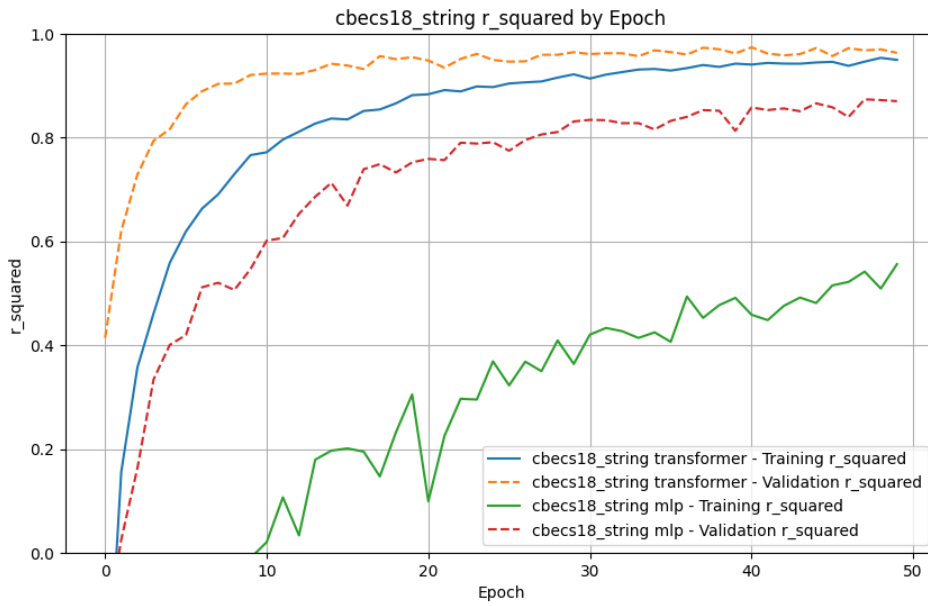


Figure 34: CBECS12 DistilBERT  $r^2$

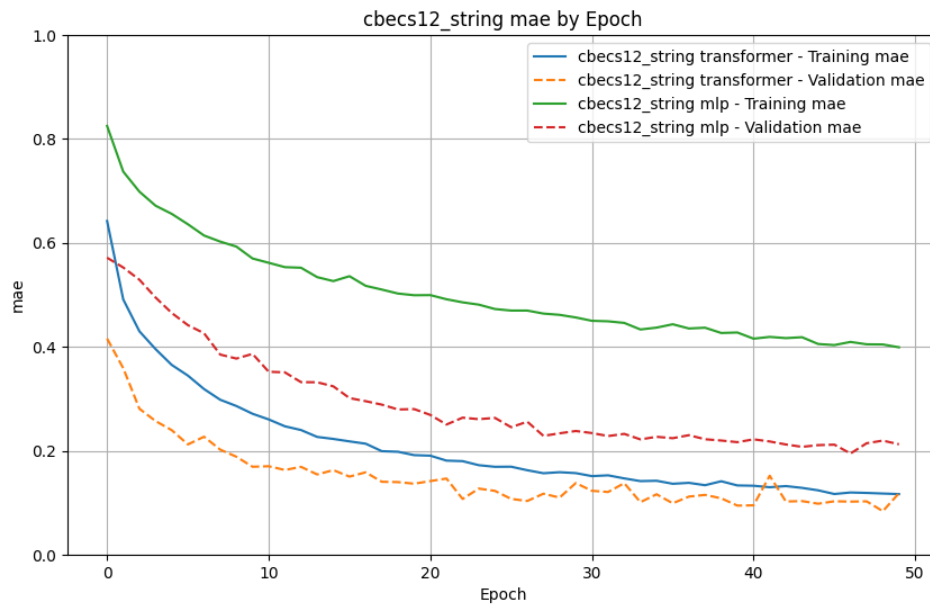


Figure 35: CBECS12 DistilBERT mae

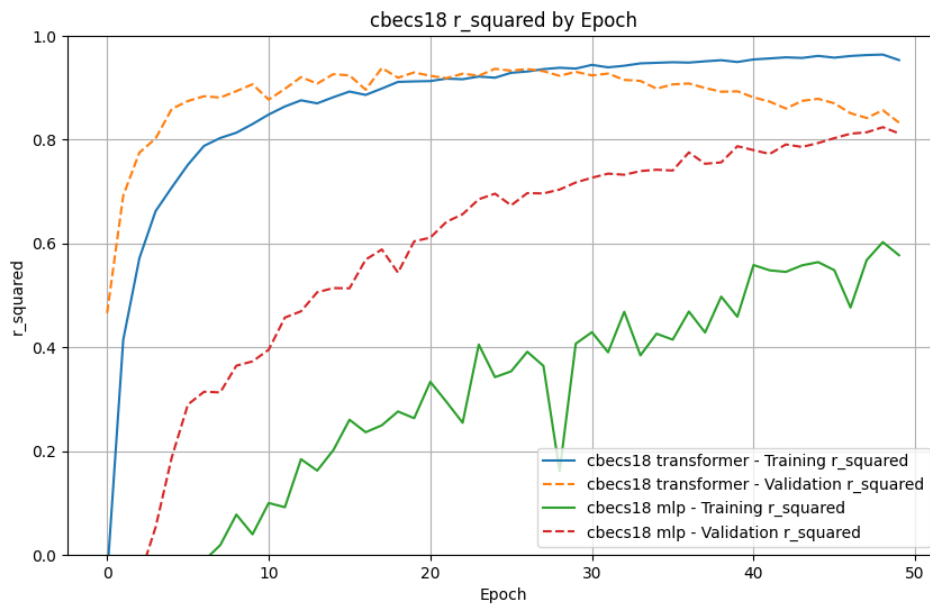


Figure 36: CBECS18 R2



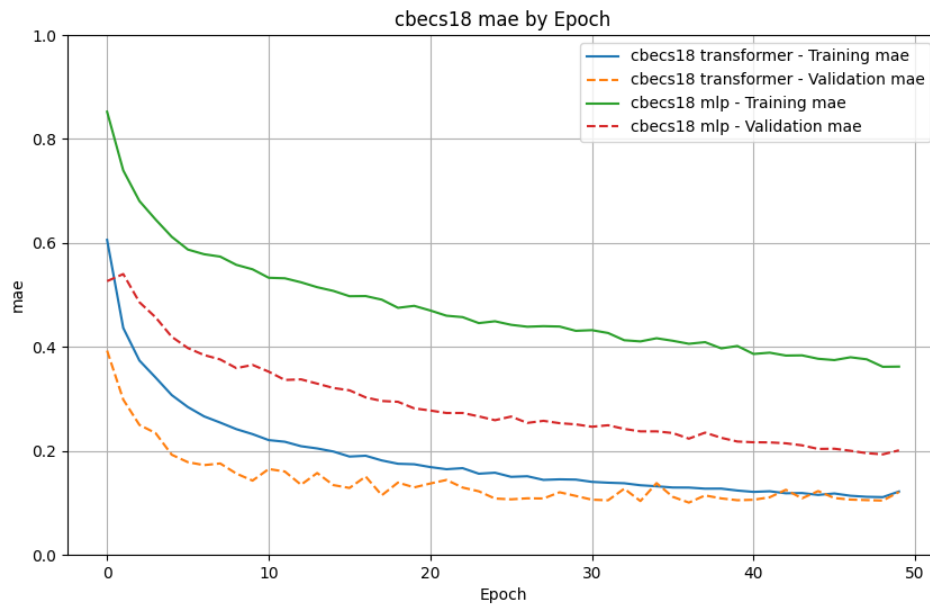


Figure 37: CBECS18 MAE

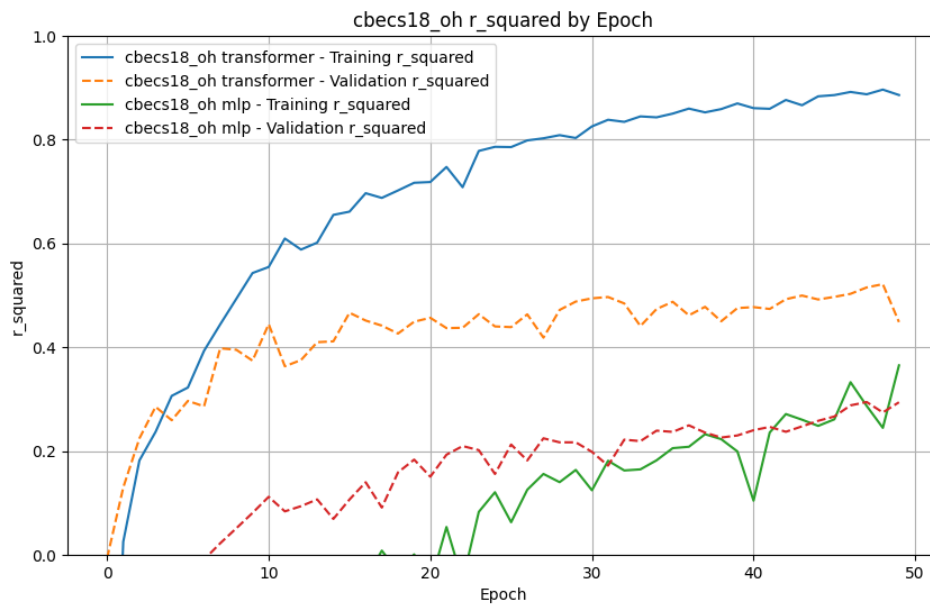


Figure 38: CBECS18 One Hot  $r^2$

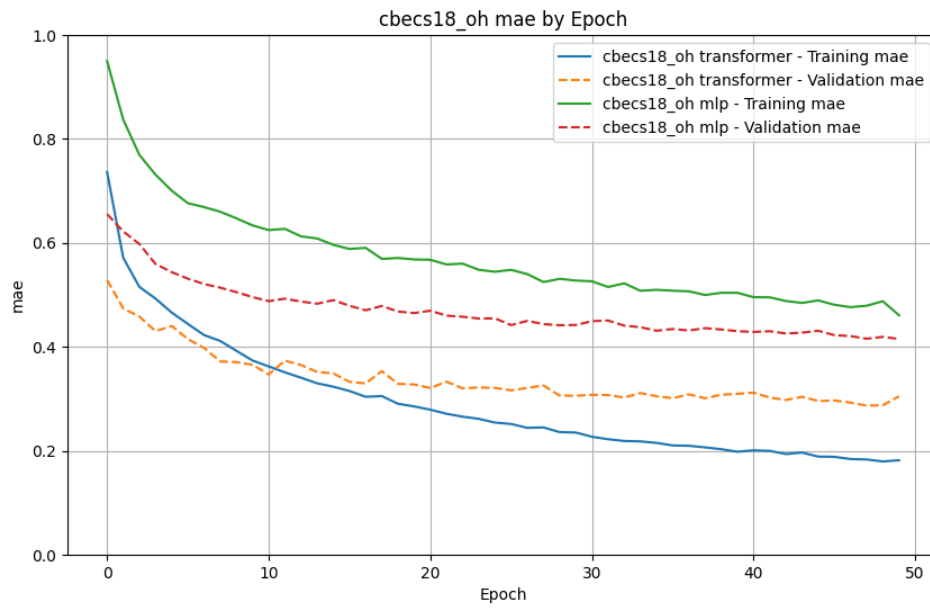


Figure 39: CBECS18 One Hot MAE

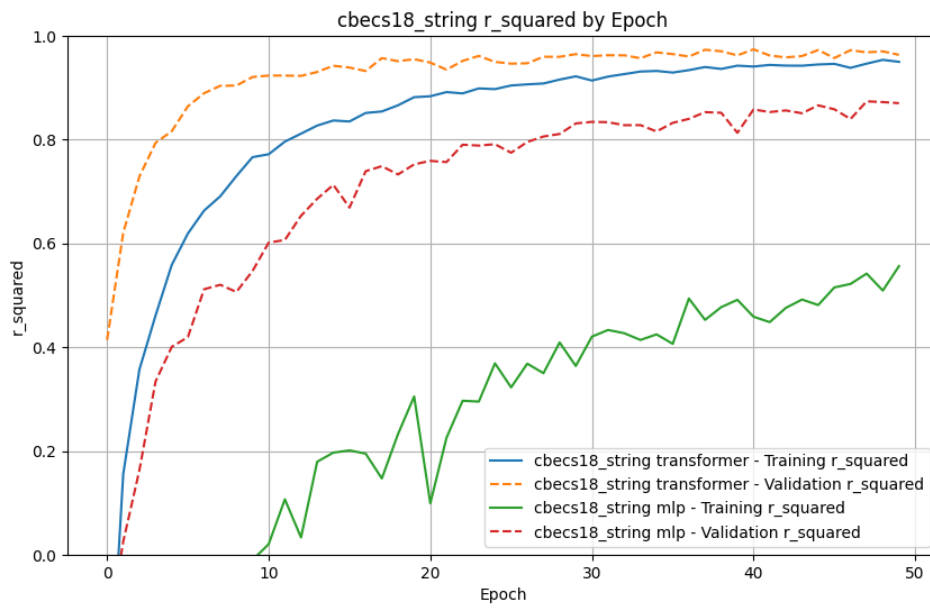


Figure 40: CBECS18 DistilBERT r2

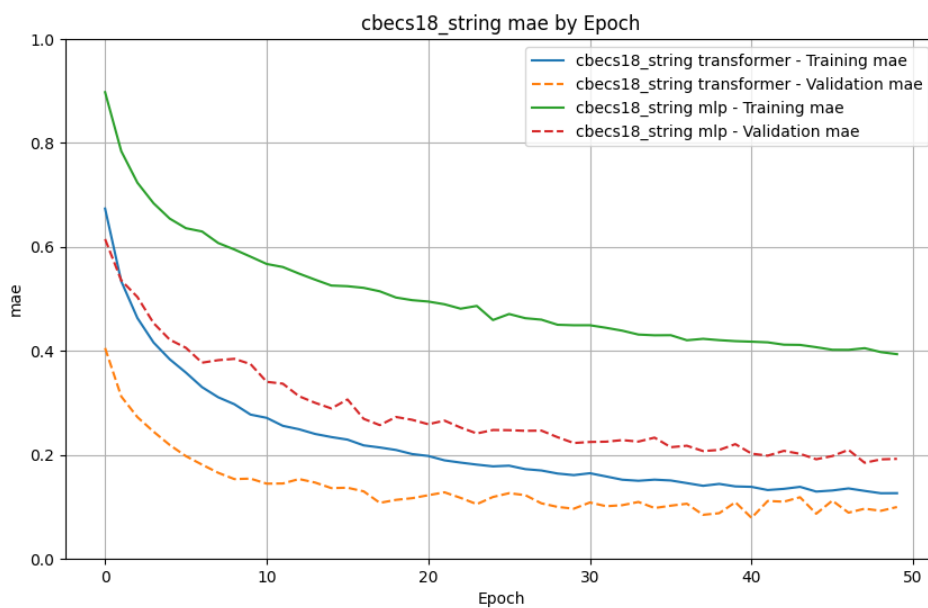


Figure 41: CBECS18 DistilBERT mae

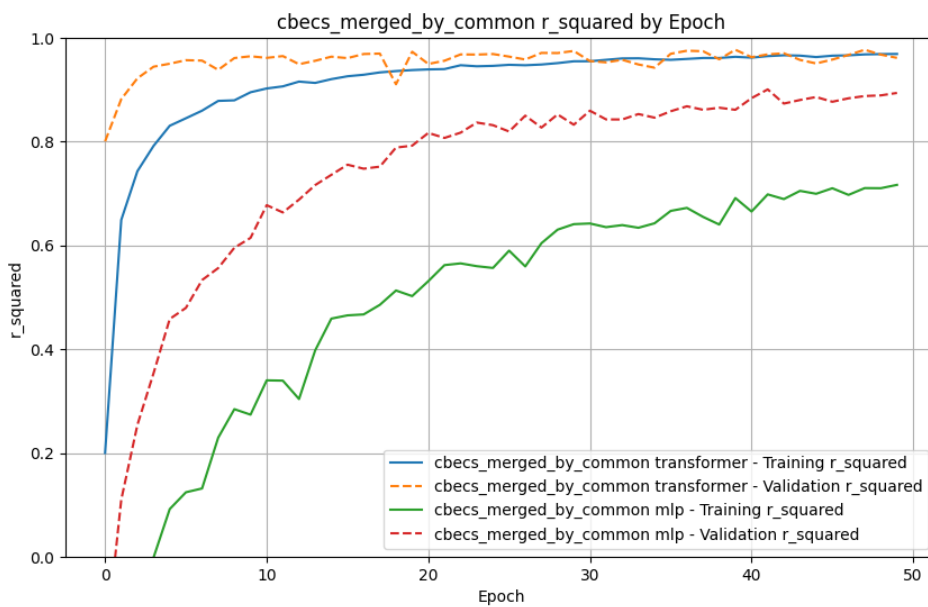


Figure 42: CBECS merged by common R2

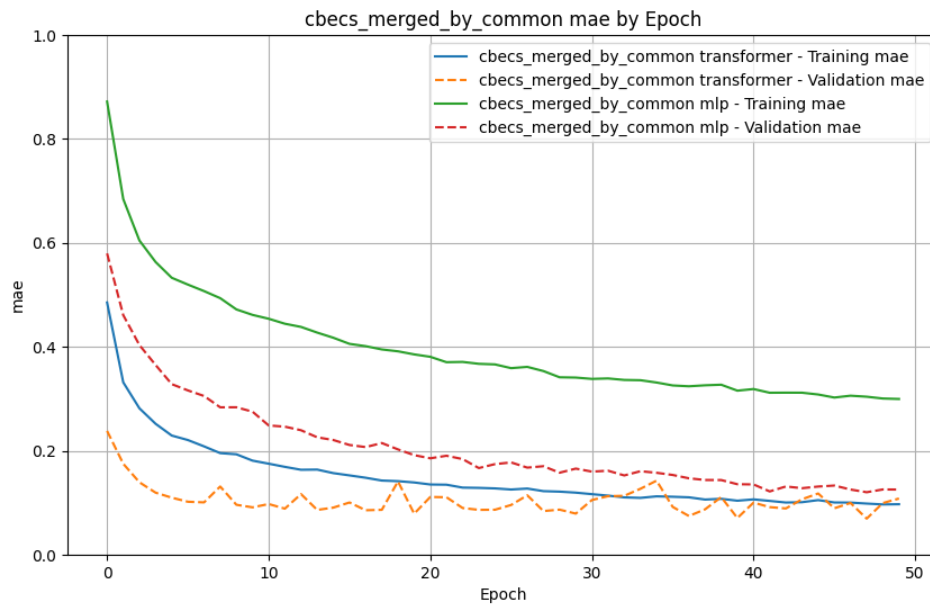


Figure 43: CBECS merged by common MAE

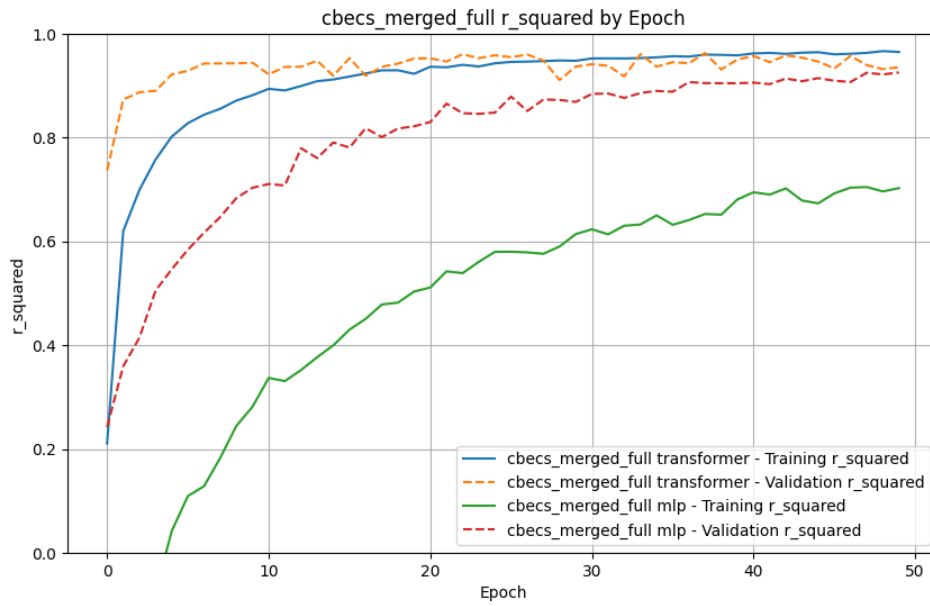


Figure 44: CBECS merged in full R2

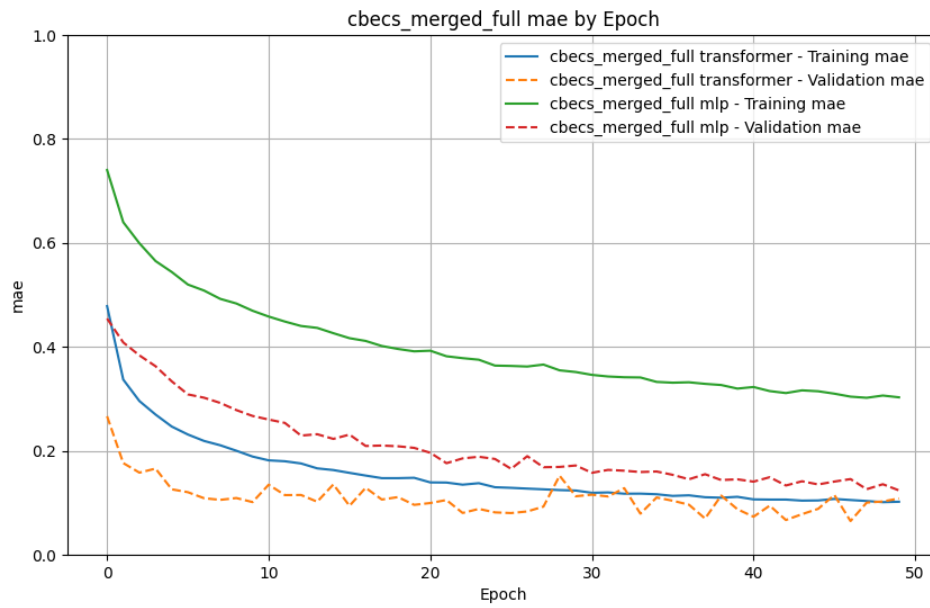


Figure 45: CBECS merged in full MAE

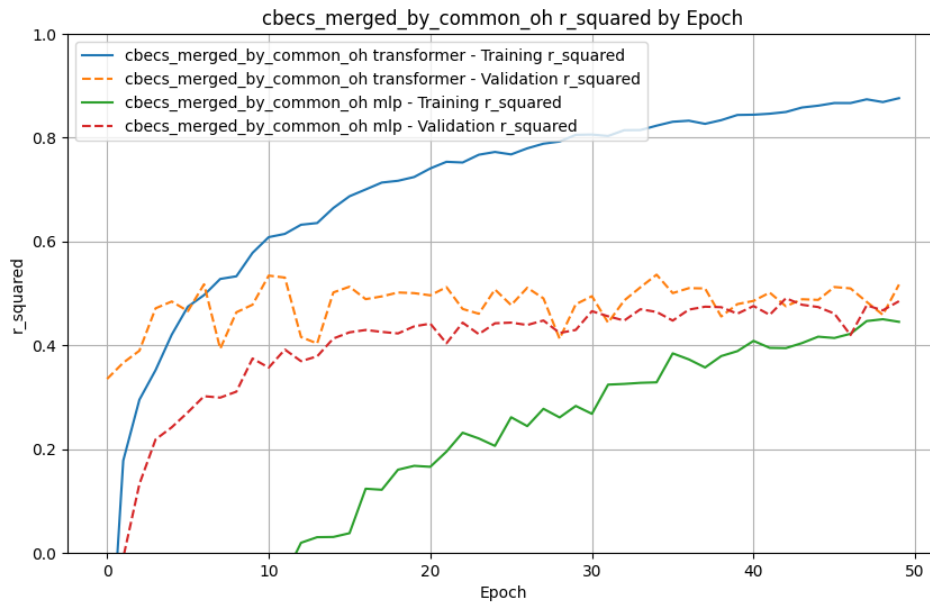


Figure 46: CBECS merged by common One Hot encoded R2

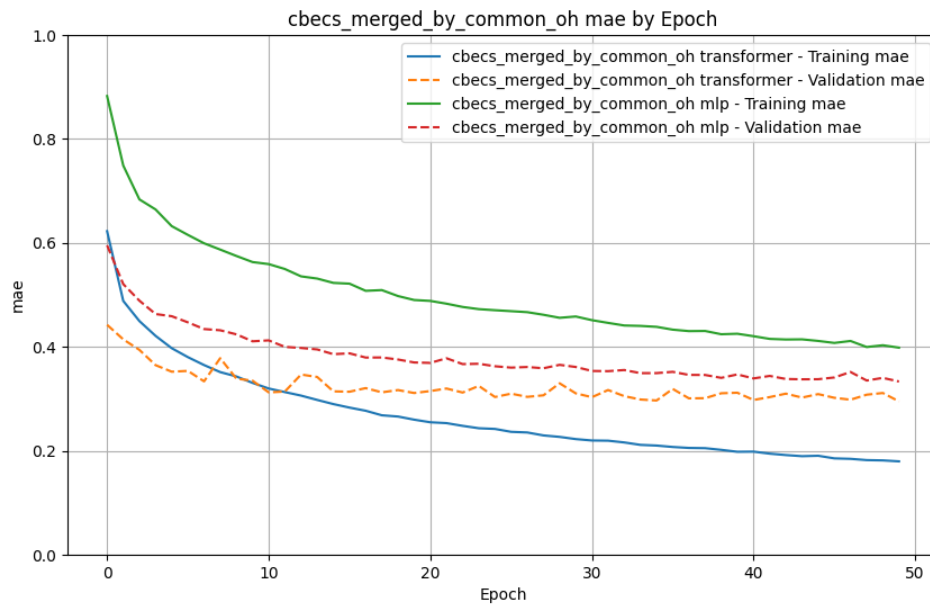


Figure 47: CBECS merged by common One Hot encoded MAE

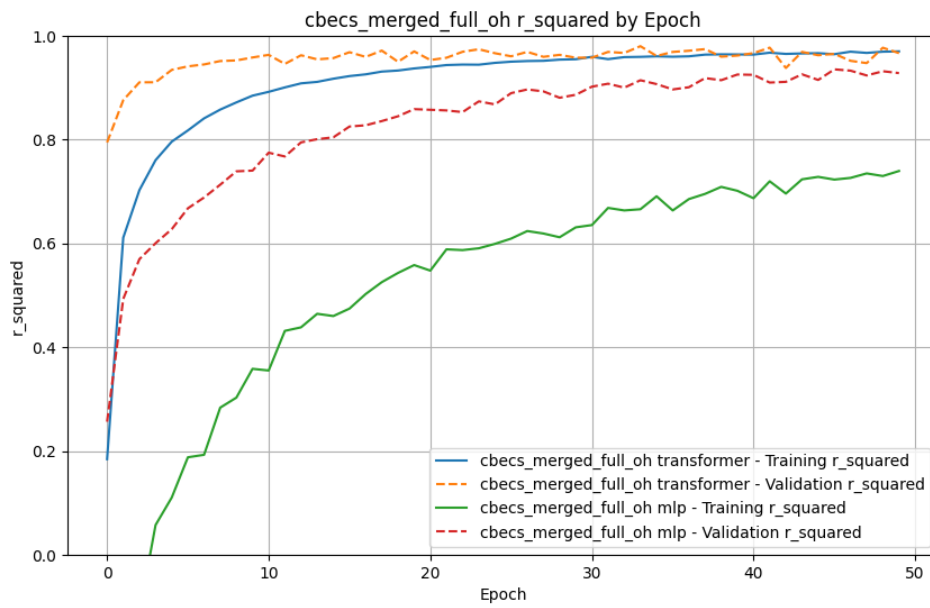


Figure 48: CBECS merged in full One Hot encoded R2

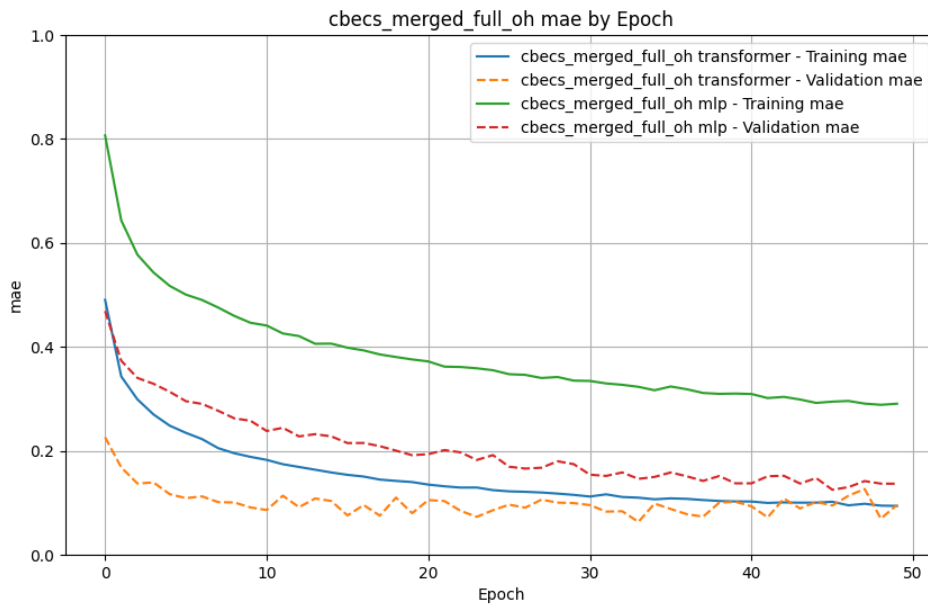


Figure 49: CBECS merged in full One Hot encoded MAE

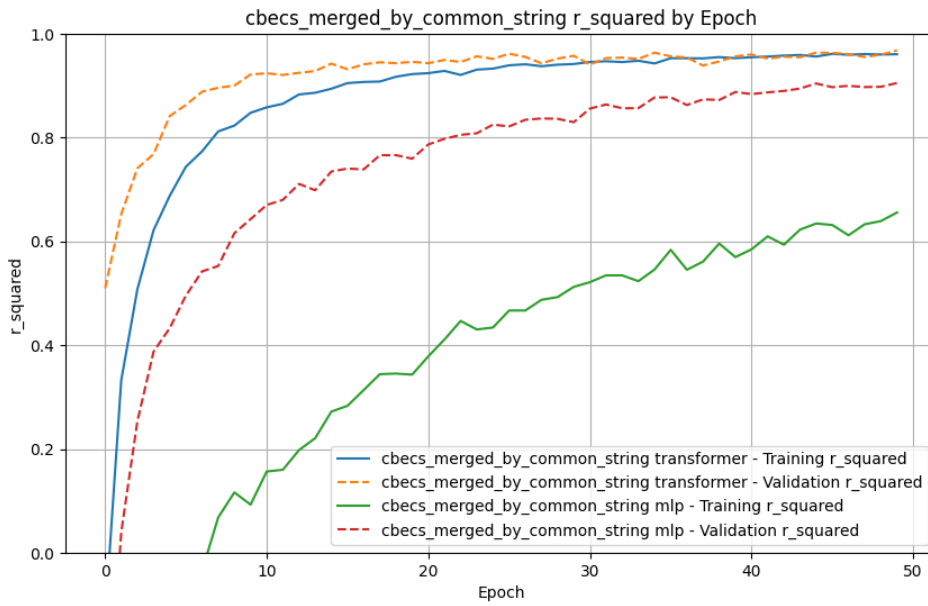


Figure 50: CBECS merged by common DistilBERT R2

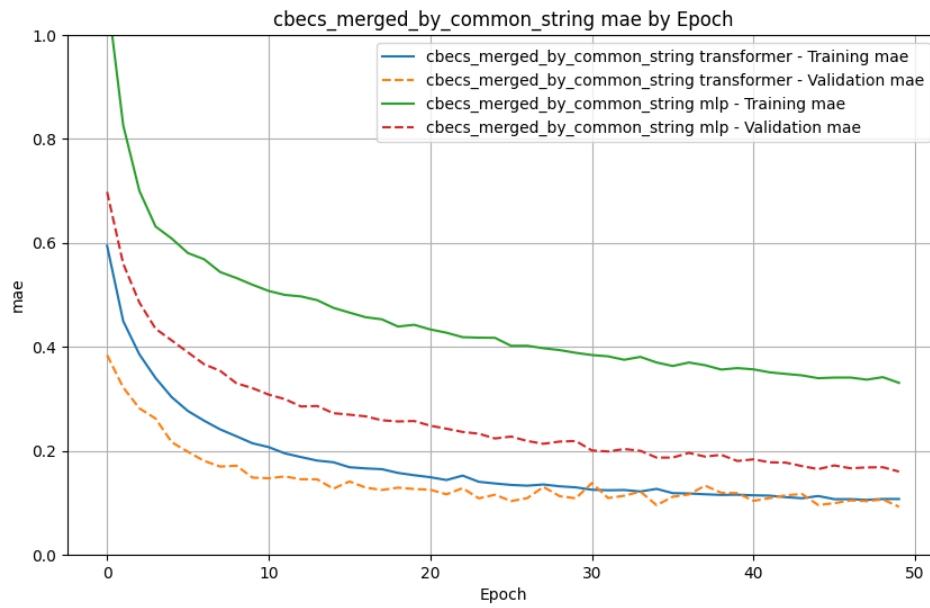


Figure 51: CBECS merged by common DistilBERT MAE

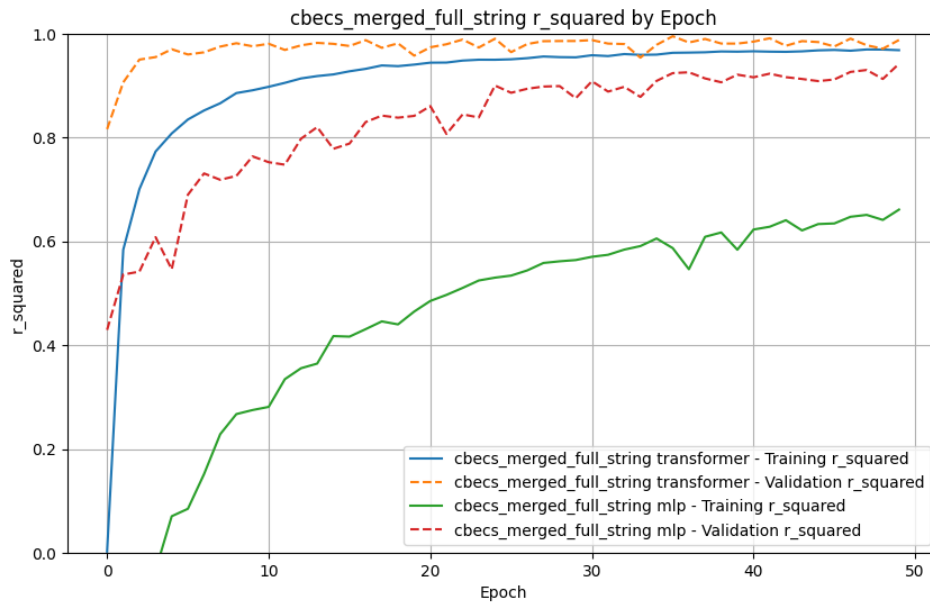


Figure 52: CBECS merged in full DistilBERT R2



## 10.2 Columns Dropped

**RECS20 Dropped Columns:** BTUELSPH, BTUELCOL, BTUELWTH, BTUELRFG, BTUELRFG1, BTUELRFG2, BTUELFZ, BTUELCOK, BTUELMICRO, BTUELCW, BTUELCDR, BTUELDWH, BTUELLGT, BTUELTREL, BTUELT1, BTUELT2, BTUELT3, BTUELAHUHEAT, BTUELAHUCOL, BTUELCFAN, BTUELDHUM, BTUELHUM, BTUELPLPMP, BTUELHTBPMP, BTUELHTBHEAT, BTUELEVCHRG, BTUELNEC, BTUELOTH, DOELSPH, DOELCOL, DOELWTH, DOELRFG, DOELRFG1, DOELRFG2, DOELFRZ, DOELCOK, DOELMICRO, DOELCW, DOELCDR, DOELDWH, DOELLLGT, DOELTREL, DOELT1, DOELT2, DOELT3, DOELAHUHEAT, DOELAHUCOL, DOELCFAN, DOELDHUM, DOELHUM, DOELPLPMP, DOELHTBPMP, DOELHTBHEAT, DOLEVCHRG, DOELNEC

**RECS15 Dropped Columns:** BTUEL, BTUELSPH, BTUELCOL, BTUELWTH, BTUELRFG, BTUELRFG1, BTUELRFG2, BTUELFZ, BTUELCOK, BTUELMICRO, BTUELCW, BTUELCDR, BTUELDWH, BTUELLGT, BTUELTREL, BTUELT1, BTUELT2, BTUELAHUHEAT, BTUELAHUCOL, BTUELEVAPCOL, BTUELCFAN, BTUELDHUM, BTUELHUM, BTUELPLPMP, BTUELHTBPMP, BTUELHTBHEAT, BTUELNEC, DOLLAREL, DOELSPH, DOELCOL, DOELWTH, DOELRFG, DOELRFG1, DOELRFG2, DOELFRZ, DOELCOK, DOELMICRO, DOELCW, DOELCDR, DOELDWH, DOELLLGT, DOELTREL, DOELT1, DOELT2, DOELAHUHEAT, DOELAHUCOL, DOLEVAPCOL, DOELCFAN, DOELDHUM, DOELHUM, DOELPLPMP, DOELHTBPMP, DOELHTBHEAT, DOELNEC

**CBECS12 and CBECS18 Dropped Columns:** ELBTU, EEXP, ELHTBTU, ELCLBTU ELVNBUT, ELWTBTU, ELLBTU, ELCKBTU ELRFBTU, ELOFBTU, ELPCBTU, ELOTBTU

## 10.3 RECS target columns

KWHAHUHEAT, KWHSPL, KWHCOR, KWHTV1, KWHPLPMP, KWHHUM, KWHFZ, KWHTV2, KWHRFG1, KWHRFG2, KWCOL, KWMICRO, KWHAHUCOL, KWHTREL, KWHDWH, KWNEC, KWHCW, KWHRFG, KWHWTH, KWHLGT, KWHDHUM, KWHCOK, KWH, KWHTBPMP, KWHTBHEAT, KWCFAN