

Efficient-Husformer: Efficient Multimodal Transformer Hyperparameter Optimization for Stress and Cognitive Loads

Merey Orazaly^{1†}, Fariza Temirkhanova¹, and Jurn-Gyu Park^{1*}

Abstract—Transformer-based models have gained considerable attention in the field of physiological signal analysis. They leverage long-range dependencies and complex patterns in temporal signals, allowing them to achieve performance superior to traditional RNN and CNN models. However, they require high computational intensity and memory demands. In this work, we present Efficient-Husformer, a novel Transformer-based architecture developed with hyperparameter optimization (HPO) for multi-class stress detection across two multimodal physiological datasets (WESAD and CogLoad). The main contributions of this work are: (1) the design of a structured search space, targeting effective hyperparameter optimization; (2) a comprehensive ablation study evaluating the impact of architectural decisions; (3) consistent performance improvements over the original Husformer, with the best configuration achieving an accuracy of 88.41 and 92.61 (improvements of 13.83% and 6.98%) on WESAD and CogLoad datasets, respectively. The best-performing configuration is achieved with the $(L + d_m)$ or $(L + FFN)$ modality combinations, using a single layer, 3 attention heads, a model dimension of 18/30, and FFN dimension of 120/30, resulting in a compact model with only ~30k parameters.

Index Terms—Hyperparameter Optimization (HPO), Multimodal Transformer, Attention Mechanisms, Wearable Sensor Data, Physiological Signal Processing, Stress Detection, Cognitive Load

I. INTRODUCTION

Transformer-based architectures have gained significant attention across a broad range of research domains in the processing of physiological signals. It has the capacity to capture long-range dependencies and extract features from complex, high-dimensional data. Since conventional Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) [1] process sequences in a local manner and are not capable of processing sequences in their entirety, self-attention-based Transformer models are therefore well-suited for performing stress detection tasks. Transformer-based models yield considerable improvements for healthcare applications such as ECG classification [2], emotion recognition [3], and monitoring based on wearables [4]. The recent literature has dealt with Transformer extensions like Time Series Transformers (TST) and Physiological Signal Transformers (PST) [5] [6], demonstrating that these models can effectively manage multimodal biosignals and enhance the accuracy of classification.

However, a big challenge is the optimization of Transformer architectures toward real-time and resource-efficient implementation. Although the Transformers were successful in

This manuscript is currently under review.

¹The School of Engineering and Digital Sciences, Nazarbayev University, Astana, Kazakhstan (*corresponding author: jurn.park@nu.edu.kz). [†]Work started as a MS thesis at Nazarbayev University.

physiological signal analysis, the challenge that exists for the design of efficient Transformer models towards the multi-class stress detection is linked to their immense computational load and memory requirements coupled with trade-offs between accuracy and efficiency. Standard Transformer architectures need a considerable amount of computational resources; therefore, they are impractical for performing real-time operations and resource-constrained platforms such as mobile and wearable devices [7]. In the case of multi-class stress detection, the model has to preserve generalization properties over various states of physiological signals while also prioritizing low latency and energy consumption [8].

Mostly, existing approaches primarily concentrate on accuracy, ignoring the practicalities of Transformer deployment in real-world contexts with hardware limitations. Addressing those problems requires systematic research on hyperparameter optimization (HPO) for Transformer models, specifically focusing on layers (L), attention heads (H), dimension (d_m) and FFN dimension (FFN) in order to provide a practical and scalable framework for stress detection tasks.

In this paper, we propose a Husformer-based [9] optimized model, a highly efficient Transformer-based model applied to multi-class stress detection using the WESAD [10] and CogLoad [11] datasets. We introduce an optimization strategy that adjusts systematically the number of parameters to reach an optimal trade-off between accuracy and efficiency.

The paper specifically highlights the following contributions:

- Propose a hyperparameter optimization strategy that employs a constrained local search space to enable efficient tuning of Transformer models (e.g., Husformer) for performance improvements in resource-constrained platforms, using small-scale affective computing and cognitive load datasets of WESAD and CogLoad.
- Perform a systematic ablation study to identify combinations of the most critical components that contribute to the model's effectiveness.
- The Efficient-Husformer achieves the accuracies of 88.41 and 92.61 on WESAD and CogLoad, respectively, corresponding to substantial improvements of 13.83% and 6.98% over the original Husformer results, with a dramatically reduced number of parameters.

Our open-source codes are on GitHub¹.

The rest of the paper is organized as follows: Section II presents the background, motivation, and summary of related works. The search space and optimization strategy are proposed in Section III. The descriptions of experimental settings,

¹<https://github.com/Merey1508/Efficient-Husformer>

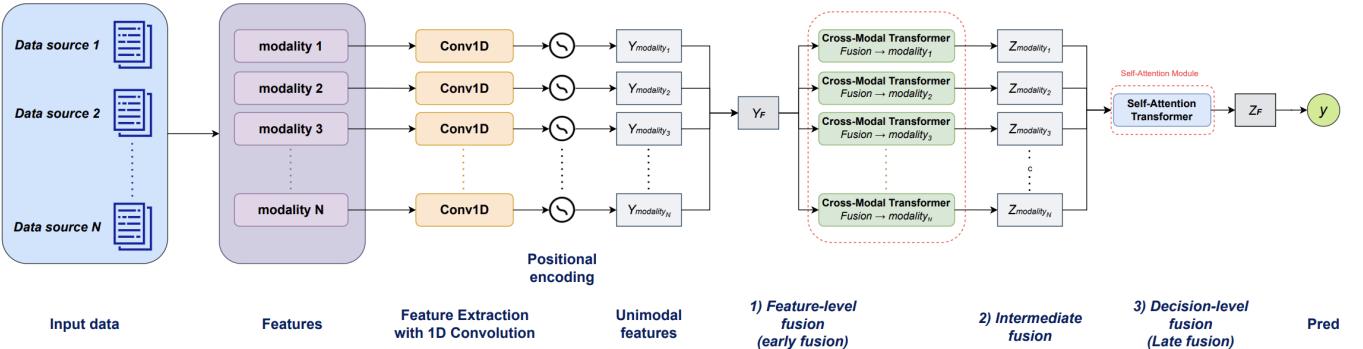


Fig. 1. Efficient-Husformer Architecture (Deployed from [9] and Modified).

datasets, evaluation, and quantitative metrics are outlined in Section IV. Section V contains summaries of the quantitative assessment, ablation study, and results and analysis. The discussion and future work are presented in Section VI, and the paper is concluded in Section VII.

II. MOTIVATION AND RELATED WORK

A. Background

The Husformer is defined by three major components, which are illustrated in Figure 1: Before Cross-Modal Transformer, Cross-Modal Transformer, and Self-Attention Transformer for final prediction.

1) *Before Cross-Modal*: In the first phase, multimodal physiological signals (e.g., GSR, BVP, EMG, ECG, RESP) are treated as discrete input streams (modalities). Each modality is processed separately through a 1D convolutional layer (Conv1D) that highlights spatial features of that signal type. After convolution, positional encoding is added to each unimodal feature vector to reflect the order in which features are extracted – which is essential for time series data. The end result is a sequence of unimodal feature vectors, one per source, in a fixed-dimensional space. The unimodal features are then concatenated together to form a joint representation, labeled Y_F . This is the early fusion or feature-level fusion, where features from all modalities are taken together before reaching the Transformer layers.

2) *Cross-Modal Transformer*: In this core module, it enriches each unimodal feature by providing contextual information from the other modalities using a Cross-Modal Transformer. The cross-modal Transformer consists of U stacked encoder layers, where each layer consists of a Multi-Head Cross-Modal Attention Block and a Position-Wise Feed-Forward Network (FFN). Given the Transformer architecture, all components are wrapped in residual connections and layer normalization. Provided a unimodal target feature sequence Y_{M_i} and a shared fused representation Y_F , the function to transform between the encoder layers is defined as follows:

$$Z_F^{[0]} = Y_F \quad (1)$$

$$Z_{M_i}^{[0]} = Y_{M_i} \quad (2)$$

$$\hat{Z}_{M_i}^{[u]} = \text{CM}_{\text{Mul}}^{[u]} \left(L(Z_F^{[0]}), L(Z_{M_i}^{[u-1]}) \right) \quad (3)$$

$$\dot{Z}_{M_i}^{[u]} = \hat{Z}_{M_i}^{[u]} + L(Z_{M_i}^{[u-1]}) \quad (4)$$

$$Z_{M_i}^{[u]} = F_\theta \left(L(\dot{Z}_{M_i}^{[u]}) \right) + L(\dot{Z}_{M_i}^{[u]}) \quad (5)$$

where $u \in \{1, \dots, U\}$ is the encoder layer index, $L(\cdot)$ is layer normalization, $F_\theta(\cdot)$ is the position-wise feed-forward network, and $\text{CM}_{\text{Mul}}^{[u]}$ is the cross-modal attention function used in the transformation step based on the fact that it uses $Z_F^{[0]}$ as keys and values, and $Z_{M_i}^{[u-1]}$ as query.

3) *Self-Attention Transformer*: The last stage of the Husformer architecture processes decision-level fusion with a Self-Attention Transformer. After cross-modal encoding, the updated modality-specific representations $\{Z_{M_i}^{[U]}\}$ are merged and passed into a Transformer module with L layers, where each layer has H attention heads along with d_{ffn} -sized feed-forward blocks. In this Transformer, global dependencies across all modality embeddings will be captured through the multiple layers, allowing the model to hone in on the joint representation Z_f by attending to complementary characteristics across modalities. The result is a context-aware, high-level feature vector used in a prediction head to generate the model output:

$$y = \text{Pred}(Z_f)$$

This way of allowing flexible interaction across the modality-specific signals help the model make better and more robust predictions by virtue of the temporal and contextual patterns learned across the other stages of the pipeline.

B. Motivation

The standard Transformer architectures are traditionally built with fixed hyperparameters such as **number of layers** (L), **attention heads** (H), **model dimension size** (d_m), and **FFN dimension size** (FFN), which may lead to sub-optimal trade-offs between accuracy and efficiency [12]. In this study, we work toward optimizing the Transformer model

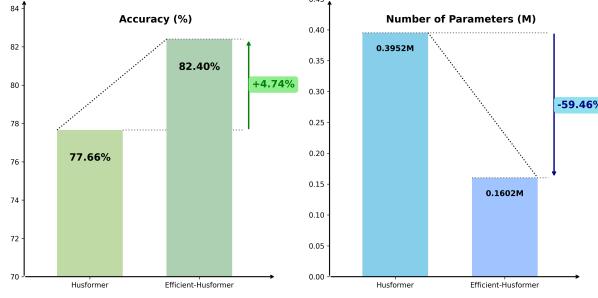


Fig. 2. Motivating Example: Comparison of Original ($L=5$) and an Efficient layer ($L = 3$) in the other default configurations: Heads ($H = 3$), Dimension size ($d_m = 30$), and Feed-Forward Network size ($FFN = 120$)

by a systematic tuning of these hyperparameters while still maintaining a competitive performance. We have defined the configurations of the Transformer models as:

$$\text{Transformers Model Variants} = f(L, H, d_m, FFN) \quad (6)$$

Motivating example: As a motivating example, we consider the baseline Husformer model, varying only a single hyperparameter — the number of layers L , which is one of the most effective in terms of performance — while keeping all other hyperparameters fixed at their default values. Surprisingly, our findings in Figure 2 show a 4.74% increase in accuracy, accompanied by a substantial reduction of 59.46% in the number of parameters, highlighting improvements in both accuracy and computational efficiency. This example provides strong empirical opportunities for re-evaluating state-of-the-art Transformer models and identifying efficient configurations better suited for multimodal datasets.

C. Related Work

Over the past years, a number of Machine Learning (ML) and Deep Learning (DL) models have been widely used in the domain of stress detection, considering different algorithms to

analyze physiological and behavioral data. As shown in Table I, we summarize our related works with the three categories of classical ML (non-DL), general DL (non-Transformers), and Transformer-based models, considering the modalities used, whether features were automatically extracted, cross-validation methods, and algorithms applied alongside their reported accuracies in classification.

1) *Classical ML (non-DL) models:* A notable study by Schmidt et al. [10] discusses the use of multimodal data for affect detection using multiple sensor modalities, such as Accelerometer (ACC), Blood Volume Pulse (BVP), Electrodermal Activity (EDA), Skin Temperature (TEMP), Respiration (RESP), Electromyogram (EMG), and Electrocardiogram (ECG), along with a wide variety of classifiers such as Decision Tree (DT), Random Forest (RF), Ada Boost (AB), Linear Discriminant Analysis (LDA), and K-Nearest Neighbors (KNN), with accuracy rates between 74.20% and 87.74% using the leave-one-subject-out (LOSO) method. Another study by Aqajari et al. [13] develops a model that uses features of pyEDA, a Python toolkit for EDA, for pre-processing and statistical and automatic feature extraction. In the binary classification task for stress recognition, their model achieves an accuracy of 79.71% on the WESAD dataset, surpassing previous approaches.

Bobade et al. [14] incorporate Support Vector Machine (SVM) and Artificial Neural Network (ANN) models, yielding higher accuracy values of 87.59–95.21%. The performance improves from increasing the number of classifiers, allowing better handling of the complexity and variability of the data between subjects. In the study of Su et al. [15], four ML algorithms were developed for college students' models, by developing stress prediction models: RF, Logistic Regression (LR), SVM, and Feed-Forward Neural Network. The RF model resulted in the best predictive capability for stress levels, with the highest performance among all models, by reaching an accuracy of 84.62%, specificity of 96.35%, and AUC and

TABLE I
SUMMARY OF RELATED WORKS

Study	Multi-class	Dataset	CV	Modalities	Auto. Extracted Features	Algorithms (Accuracy)
Classical ML Models						
Schmidt et al. (2018) [10]	✓	WESAD	✗	7	✗	DT, RF, AB, LDA, KNN (74.20%–87.74%)
Aqajari et al. (2021) [13]	✗	WESAD	LOOCV	1	✓	KNN, RF, SVM, NB (85%–91.60%)
Bobade et al. (2020) [14]	✓	WESAD	K-fold (k=5)	7	✗	DT, RF, AB, LDA, KNN, SVM, ANN (87.59%–95.21%)
Su et al. (2022) [15]	✓	WESAD	K-fold (k=10)	3	✗	RF, LR, SVM, FNN (84.62%)
DL Models						
Behinaein et al. (2021) [2]	✗	ECG	✗	1	✓	Transformers (80.4%)
Yao et al. (2021) [16]	✓	Custom	K-fold (k=5)	2	✓	MUSER Transformer (84.2%)
Ziaratnia et al. (2023) [17]	✓	Custom	✗	1	✓	CCT-LSTM (83.2%)
Husformer. (2024) [9]	✓	WESAD	✗	6	✓	Transformers (78.68%)

F1 of 82%. Their work highlighted the importance of specific modalities in achieving good performance while maintaining the complexity of the model as low as possible.

2) *DL models*: Attention mechanisms have further enhanced the extraction of relevant features by assigning varying importance to different data segments [18], which is crucial in identifying critical moments of stress-related physiological changes. Using this approach, Behinaein et al. [2] presented only ECG data, using Transformers for feature extraction, and reported an accuracy of 80.4% by employing the LOSO cross-validation technique. Yao et al. [16] proposed MUSER, a Transformer-based model whose performance in the task of detecting stress was facilitated by emotion recognition as an auxiliary task. Consequently, MUSER relied on the interdependence between the two variables (stress and emotion) and achieved 84.2% accuracy results in the Multimodal Stressed Emotion-MuSE dataset, an indication of the multi-task learning benefit in performing affective computing. These studies further point out that deep learning models improve the process of stress detection by using multimodal data and sophisticated architectures. Ziaratnia et al. [17] proposed a novel method on remote video-based stress estimation using a Convolutional Channel-wise Transformer combined with Long Short-Term Memory (CCT-LSTM). Their approach provided better performance on the spatial and temporal features extracted from facial cues, yielding an accuracy of 83.2% and an F1 score of 83.4%.

Finally, Wang et al. [9], an end-to-end multimodal Transformer named Husformer (as our reference model), which learns from multimodal data streams. Utilizing cross-modal attention and self-attention transformers, it effectively identifies relevant contextual information and captures long-term temporal dependencies in human affective states. Husformer consistently outperforms the state-of-the-art multimodal baselines and single-modality methods, including datasets without further feature engineering. Specifically, on WESAD and CogLoad, it achieved accuracy results of 78.68% and 74.06%, respectively, revealing superior adaptivity and capacity for meaningful and effective pattern recognition.

III. METHODOLOGY

This section outlines the essential search space and optimization strategy that conceptualize and develop efficient Transformer models applied to different datasets, searching for efficient Transformer architectures that maximize multimodal learning without sacrificing task-specific performance.

Taking into account the multimodal characterization of their physiological signals, we further introduce a decoupled optimization strategy in which the cross-modal Transformer (L_{cm}, H_{cm}) and the self-attention Transformer (L_{sa}, H_{sa}) for modality fusion are separately optimized:

$$f(L_{cm}, H_{cm}) + f(L_{sa}, H_{sa}) + f(d_m) + f(FFN) \quad (7)$$

Therefore, this parametric configuration allows independent optimizations for both modality fusion and classification functionalities for improved efficiency and interpretability. To

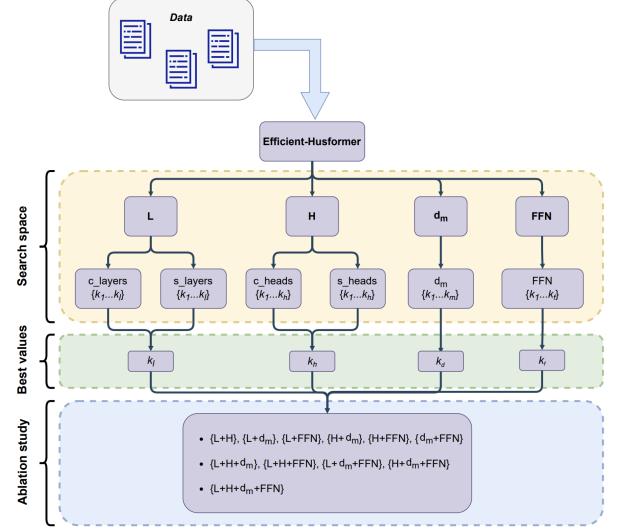


Fig. 3. Methodology Overview

TABLE II
SEARCH SPACE

Model Size		
Attention	Layers	$\{k_1, k_2, \dots, k_l\}$
	Heads	$\{k_1, k_2, \dots, k_h\}$
	Model Dimension	$\{k_1, k_2, \dots, k_m\}$
FFN	FFN Size	$\{k_1, k_2, \dots, k_f\}$

effectively navigate this optimization space and strategy, we aim to address the following three questions based on the three steps (Figure 3):

- How can a structured search space combined with local optimization strategies guided by priority assumptions improve the efficiency and effectiveness of hyperparameter optimization in Transformer-based models?
- Which architectural and training components identified from a systematic ablation study are the most important to the performance of the Transformer model?
- To what extent do the proposed Efficient-Husformer models outperform the original Husformer in terms of classification accuracy, and does its open-source implementation facilitate reproducibility?

With these three questions, we establish a comprehensive optimization framework that balances performance, resource efficiency, and deployment feasibility. The study characterizes the influence of components of the Transformer on tasks for stress detection and offers further guidance on efficient design for Transformer models.

A. Search Space

For the optimization space, as shown in Table II, we simultaneously focus on key architectural hyperparameters to balance computational efficiency and representational power. For each hyperparameter, we access a smaller but still expressive range of values from the baseline configurations. The constraint of hyperparameters objectively allows us to

systematically and reproducibly optimize, without generating exhaustive enumerations of parameters.

More importantly, we believe the parameter space captures the optimal configurations for maximizing efficiency (latency, memory, and energy consumption). This parameter space also enables a variety of architectural variants that are still expressive enough to model multimodal dynamics (e.g., stress detection) across several datasets.

B. Optimization Strategy

As shown in Figure 3, the optimization process follows a rigorous three-step process:

- 1) **Defining the search space:** In the first step, we establish a search space that is bounded and meaningful for the model architecture and the data specifics. Each hyperparameter is given a finite set of discrete values.
- 2) **Identifying the best configurations:** In the second step, we have candidate combinations that are marked according to a trade-off between predictive performance and computational cost. Candidate models with the best performance in this stage are selected for the next tuning phase of the study.
- 3) **Ablation study:** In the final step of the process, we conduct a systematic ablation study to fully understand the contribution of the individual and combined contributions of the core components of the structure. We conduct experiments where we vary or eliminate one component at a time to see the contribution of the core component. We also test all combinations of pairs (e.g., $(L + H)$, $(H + d_m)$), triples (e.g., $(L + H + FFN)$), and the full combination (all four modules).

This exhaustive approach allows us to examine the individual contributions of the components through additive and interaction effects. Importantly, the search space is constructed in a way that maintains architectural constraints; for example, d_m values must be divisible by H for multi-head attention components. This constraint-based design minimizes the space to only those computationally valid parameter settings, while reducing the chance of invalid model generations occurring. Therefore, only operationally sensible and computationally feasible parameter space for optimization is used.

Specifically, the model dimension (d_m) serves a dual purpose: it defines the size of the token embedding dimensions and simultaneously acts as the projection space for the attention mechanism; therefore, the feed-forward network (FFN) is defined by two linear layers, and it is tied to the model dimensions through a scale factor, α , resulting in the hidden size of FFN as $(\alpha \times d_m)$. The coupling of these metrics indicates that tuning the FFN, along with the model dimension, is crucial in maintaining consistency and allowing for valid projections across layers. On the other hand, the number of attention heads and the number of Transformer layers on both self-attention and cross-modal branches can be considered relatively independent.

In order to progressively evaluate the relative importance of each hyperparameter in the Efficient-Husformer architecture, we conduct a simple yet effective hyperparameter optimization

TABLE III
SPECIFICATIONS OF EXPERIMENTAL PLATFORM

Laptop Platform	
GPU	2304-core NVIDIA GeForce RTX 2070, 8GB VRAM, CUDA 12.7, NVIDIA-SMI 566.36
CPU	Intel-based processor
Memory	16GB RAM
Operating System	Microsoft Windows 10
Power	3W (idleness) to 41W (peak load)
Software Frameworks	
Programming Language	Python 3.9
Deep Learning Framework	PyTorch 1.8

(i.e., parameter isolation) approach. Through this approach, we measure the performance difference for a hyperparameter by varying its value while keeping all other parameters at their default settings [19]. This strategy allows for the direct assessment of individual hyperparameter contributions to model accuracy, computational efficiency, and robustness when applied to multimodal stress detection on the WESAD and CogLoad datasets. The default settings provide a reliable baseline, which allows us to assume that the performance change is solely initiated by the hyperparameter taken into consideration. Each hyperparameter is tested through a valid range of values, discussed in the search space section.

IV. EXPERIMENTAL SETUP

A. Experimental Settings

Hardware Platform: All experiments were conducted on an 8GB VRAM NVIDIA GeForce RTX 2070 GPU-enabled laptop, which is supported by Microsoft Windows 10, and equipped with CUDA 12.7 and NVIDIA-SMI driver version 566.36. The GPU consumes power from 3W (idleness) to 41W (peak load). The system includes an Intel-based processor and 16GB of RAM, which are sufficient to conduct the computations for training and testing deep learning models.

Software Frameworks: We run our software on Python 3.9 and implement deep learning models using GitHub codes [?] based on the PyTorch 1.8 framework. CUDA is used to enable GPU acceleration for parallel processing.

B. Datasets

The study makes use of two datasets: Wearable Stress and Affect Detection (WESAD) [10] and Cognitive Load (CogLoad) [11].

WESAD dataset: It is a popular open-source multimodal stress and emotion recognition dataset within the fields of physiological and affective computing. The dataset comprises multimodal physiological sensor data recorded from 15 subjects, with each subject experiencing a controlled experiment for inducing various affective states, such as neutral, stress, amusement, and baseline states. Data were collected with two wearable sensors: RespiBAN, a chest-worn sensor that delivers Galvanic Skin Response (GSR), Respiration (RESP), Electrocardiogram (ECG), and Electromyogram (EMG); and Empatica E4, a wrist-worn sensor that records wrist-based GSR and Blood Volume Pulse (BVP).

To facilitate model input, these signals were segmented into fixed-size windows (700 samples for chest signals and 64 or 4 samples for wrist signals), with each segment reshaped to structured input formats suitable for model training. Only segments with consistent labels across the entire window were retained, and segments with ambiguous labeling (i.e., multiple labels within a window) were discarded. Furthermore, to focus on meaningful emotional states, segments corresponding to neutral labels were filtered out.

CogLoad dataset: The dataset [11] is the first dataset that enables the analysis of the cognitive load associated with six different tasks based on an individual's physiological responses and personality traits. It is an open-source dataset that is ready-to-use in .pk1 file format obtained from [9]. This dataset contains physiological signals (EEG, GSR, BVP), collected from 23 participants using a Microsoft Band. Each participant completed a rest session and six dual-task tasks, all of which imposed cognitive load. The first task of the dual-task was randomly selected from standard psycho-physiological tasks from the work of Haapalainen et al. [20].

C. Evaluation

Following the original Husformer evaluation practice [?], we adopted a modified 10-fold cross-validation (CV) strategy. This approach enables robust model validation by systematically rotating training, validation, and testing splits across the data. Given N samples, the data are partitioned into 10 approximately equal-sized subsets or *folds*. For each fold $i \in \{0, 1, \dots, 9\}$, the following procedure is executed:

- **Validation Set:** The i -th fold, corresponding to indices from $i \cdot \frac{N}{10}$ to $(i+1) \cdot \frac{N}{10}$, is designated as the validation set.
- **Test Set:** The subsequent $(i+1)$ -th fold is used as the test set. For the last fold ($i = 9$), the test set wraps around and includes the first $\frac{N}{10}$ samples, preserving fold size consistency.
- **Training Set:** All remaining samples not included in the current validation or test sets are assigned to the training set.

This yields non-overlapping training, validation, and test sets for each of the 10 folds, with each sample participating in all three roles (training, validation, testing) across the full cross-validation cycle.

In the `WESAD()` function, this procedure is implemented using slicing operations on the shuffled index array. For each fold, the respective sample indices for training, validation, and testing are passed to the `pk1_make()` function. This function constructs and serializes the fold-specific datasets into separate .pk1 files. Each .pk1 file contains input data from six physiological modalities, along with corresponding labels and sample identifiers. This setup enables a consistent evaluation of the model's performance across multiple independent folds, with the final performance metric averaged over all folds to reduce the risk of overfitting or sample-specific bias.

D. Quantitative Metrics

Our evaluation metrics from classification and computational perspectives are as follows: average multi-class accuracy

(Acc), average multi-class F1 score (F1), cross-entropy loss (CE), mean absolute error (MAE), training duration (in hours), memory usage (in MB), and total trainable parameters. Let n be the number of classes. The average multi-class accuracy is defined as:

$$Acc = \frac{1}{n} \sum_{i=1}^n accuracy_i \quad (8)$$

where $accuracy_i$ is the binary accuracy for class i , given by:

$$accuracy_i = \frac{TP_i + TN_i}{length_i} \quad (9)$$

Here, TP_i and TN_i denote the *true positive* and *true negative* predictions for class i , respectively, and $length_i$ refers to the number of samples in that class.

The macro-averaged F1 score is computed using:

$$F1 = \frac{1}{n} \sum_{i=1}^n f1_i \quad (10)$$

where $f1_i$ is the F1 score for class i , defined as the harmonic mean between $precision_i$ and $recall_i$, precision and recall for class i :

$$f1_i = \frac{2 \cdot precision_i \cdot recall_i}{precision_i + recall_i} \quad (11)$$

Precision and recall are given by:

$$precision_i = \frac{TP_i}{TP_i + FP_i}, \quad recall_i = \frac{TP_i}{TP_i + FN_i} \quad (12)$$

where FP_i and FN_i are the number of *false positive* and *false negative* predictions for class i , respectively.

In terms of loss-based evaluation, the cross-entropy loss CE is additionally included, defined over all N samples and n classes as:

$$CE = -\frac{1}{N} \sum_{j=1}^N \sum_{i=1}^n y_{ji} \log(\hat{y}_{ji}) \quad (13)$$

where y_{ji} is the ground truth label and \hat{y}_{ji} is the predicted probability for class i on sample j .

We also report the mean absolute error MAE , a general-purpose metric, indicating average prediction error magnitude:

$$MAE = \frac{1}{N} \sum_{j=1}^N |y_j - \hat{y}_j| \quad (14)$$

where y_j and \hat{y}_j are the ground truth label and the predicted probability for the sample j , respectively.

To supplement the classification metrics, we also report the computational cost. Specifically, we provide the total training time T_{train} , defined as:

$$T_{train} = \sum_{e=1}^E t_e \quad (15)$$

where t_e indicates the elapsed time for the epoch e , and E is the total number of training epochs.

Memory efficiency is measured through maximum memory consumption M_{peak} at training time t :

$$M_{peak} = \max_{t \in [0, T_{train}]} memory(t) \quad (16)$$

TABLE IV
LAYERS CONFIGURATION RESULTS ON WESAD AND COGLOAD DATASETS (DEFAULT: $c_layers = s_layers = 5$, $c_heads = s_heads = 3$, $d_m = 30$, $FFN = 120$); THE BEST RESULTS ARE SHOWN IN **bold**, AND THE WORST RESULTS ARE UNDERLINED)

No	Cross-Modal Layers	Self-Attention Layers	Loss	MAE	Accuracy	F1	Train Time (h)	Memory (MB)	Params
WESAD									
1	1	1	0.2661	0.1266	0.8951	0.8955	0.45	653.68	81,910
2	2	2	0.4868	0.2320	0.8240	0.8165	0.75	1157.89	160,240
3	3	3	0.4682	0.2232	0.8295	0.8125	1.09	1652.86	238,570
4	4	4	0.5166	0.2549	0.8053	0.8015	1.36	2155.12	316,900
<u>5</u>	<u>5</u>	<u>5</u>	<u>0.5862</u>	<u>0.2715</u>	<u>0.7766</u>	<u>0.7787</u>	<u>1.74</u>	<u>2652.86</u>	<u>395,230</u>
CogLoad									
1	1	1	0.0413	0.0217	0.9643	0.9639	0.31	420.51	57,398
2	2	2	0.0645	0.0353	0.9457	0.9509	0.56	780.12	113,348
3	3	3	0.0633	0.0292	0.9460	0.9410	0.79	1132.78	169,298
4	4	4	0.0684	0.0365	0.9436	0.9382	1.06	1485.34	225,248
<u>5</u>	<u>5</u>	<u>5</u>	<u>0.0695</u>	<u>0.0375</u>	<u>0.9427</u>	<u>0.9378</u>	<u>1.41</u>	<u>1837.90</u>	<u>281,198</u>

Additionally, we count the number of trainable parameters N_{params} to account for model complexity:

$$N_{\text{params}} = \sum_{l=1}^L P_l \quad (17)$$

where P_l refers to the number of parameters in the layer l , and L is the total number of layers.

V. RESULTS AND ANALYSIS

A. Quantitative Assessment of Key Hyperparameters

Based on all of the experiments conducted in this study, we organize an evidence-based order for modifying the Transformer architecture: the order is to change from the attention layers (L) and heads (H), the model dimension (d_m), and finally to the FFN hidden size (FFN). This order is driven by empirical performance and cost.

1) *Layer Effects*: Table IV presents the results of varying the number of cross-modal (L_{cm}) and self-attention (L_{sa}) layers from 1 to 5, with other configurations fixed at the default. For both datasets, surprisingly increasing the number of layers leads to consistent performance degradation. On the WESAD dataset, the configuration with a single layer in both the cross-modal and self-attention blocks achieves the highest accuracy (89.51%) and F1 score (89.55%), requiring significantly less computation and enabling faster training. As the depth increases, the accuracy steadily drops to 77.66% with the five-layer architecture. A similar trend is observed on the CogLoad dataset, where the one-layer model attains the accuracy of 96.43%, outperforming those of deeper models, in which the five-layer drops to 94.27%.

As shown in Figure 4, we compare the training and validation error patterns on WESAD and CogLoad datasets. The 5-layer Transformers demonstrate the patterns of overfitting trend, a phenomenon commonly associated with increasing model depth in DL architectures. This tendency is particularly evident when applied to small and structured datasets. The phenomenon of diminishing returns indicates that the addition of more layers primarily amplifies noise in the data rather than contributing to the extraction of meaningful hierarchical features and latent patterns [21] [22]. As a result, the gap between training and validation performance widens; while

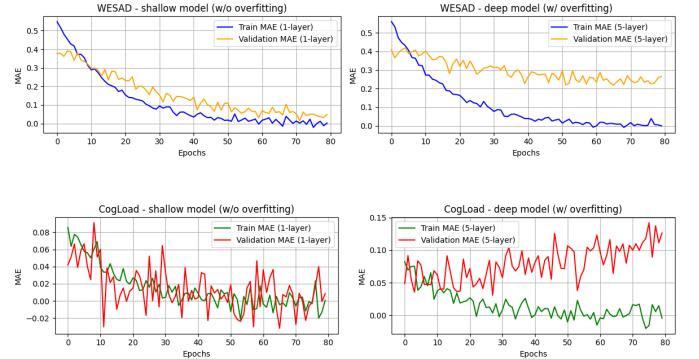


Fig. 4. Comparison of Training and Validation Errors during Training between 1-layer (left) and 5-layers (right) on WESAD (top) and CogLoad (bottom) datasets

training error decreases, validation error stagnates on WESAD or rises on CogLoad in later epochs.

2) *Head Effects*: Table V explores the performance results of changing the number of attention heads in both Transformer modules, while keeping other parameters fixed. The results indicate that a single attention head in each module yields the best performance across both datasets. On WESAD, increasing the number of heads from one to three results in a noticeable drop in accuracy, from 81.96% to 77.66%, accompanied by a significant increase in memory usage. Similarly, the CogLoad exhibits a decline in performance as the number of heads increases.

The observed performance degradation with additional heads is likely due to feature space fragmentation. With small hidden dimensions (e.g., $d_m = 30$), multi-head attention divides the limited representation into even smaller subspaces, weakening the expressiveness of each head. Since both datasets consist of relatively low-dimensional and temporally stable signals, the complexity added by multiple attention heads becomes redundant and introduces variance that harms generalization [23] [24] [25].

3) *Dimension Size Effects*: As shown in Table VI, we evaluate the sensitivity of the model to variations in hidden dimension size d_m by testing three configurations: $d_m = 9$, 18, and 30. On the WESAD dataset, the configuration with $d_m = 18$ achieves the best accuracy (79.32%), outperforming

TABLE V

HEADS CONFIGURATION RESULTS ON WESAD AND COGLOAD DATASETS (DEFAULT: $c_layers = s_layers = 5$, $c_heads = s_heads = 3$, $d_m = 30$, $FFN = 120$). THE BEST RESULTS ARE SHOWN IN **bold**, AND THE WORST RESULTS ARE UNDERLINED.

No	Heads (H_{cm})	Heads (H_{sa})	Loss	MAE	Accuracy	F1	Train Time (h)	Memory (MB)	Params
WESAD									
1	1	1	0.4629	0.2434	0.8196	0.8116	1.42	1583.21	395,230
2	2	2	0.5552	0.2830	0.7820	0.7796	1.51	2091.66	395,230
<u>3</u>	<u>3</u>	<u>3</u>	<u>0.5862</u>	<u>0.2715</u>	<u>0.7766</u>	<u>0.7787</u>	<u>1.74</u>	<u>2652.86</u>	<u>395,230</u>
CogLoad									
1	1	1	0.0532	0.0279	0.9542	0.9527	1.39	1432.47	281,198
2	2	2	0.0668	0.0381	0.9360	0.9348	1.37	1635.81	281,198
<u>3</u>	<u>3</u>	<u>3</u>	<u>0.0695</u>	<u>0.0375</u>	<u>0.9427</u>	<u>0.9378</u>	<u>1.41</u>	<u>1837.90</u>	<u>281,198</u>

TABLE VI

DIMENSION SIZE CONFIGURATION RESULTS ON WESAD AND COGLOAD DATASETS (DEFAULT: $c_layers = s_layers = 5$, $c_heads = s_heads = 3$, $d_m = 30$, $FFN = 120$); THE BEST RESULTS ARE SHOWN IN **bold**, AND THE WORST RESULTS ARE UNDERLINED)

No	Dimension Size (d_m)	Loss	MAE	Accuracy	F1	Train Time (h)	Memory (MB)	Params
WESAD								
1	9	0.6780	0.3604	0.7373	0.7152	1.52	1993.34	39,154
2	18	0.4896	0.2597	0.7932	0.7932	1.61	2289.62	146,290
3	30	0.5111	0.2658	0.7767	0.7787	1.74	2652.86	395,230
CogLoad								
1	9	0.0863	0.0526	0.9183	0.9114	1.38	782.33	103,706
2	18	0.0595	0.0361	0.9351	0.9288	1.46	1315.42	189,698
3	30	0.0695	0.0375	0.9427	0.9378	1.41	<u>1837.90</u>	281,198

TABLE VII

FFN DIMENSION SIZE CONFIGURATION RESULTS ON WESAD AND COGLOAD DATASETS (DEFAULT: $c_layers = s_layers = 5$, $c_heads = s_heads = 3$, $d_m = 30$, $FFN = 120$); THE BEST RESULTS ARE SHOWN IN **bold**, AND THE WORST RESULTS ARE UNDERLINED)

No	FFN Size	Loss	MAE	Accuracy	F1	Train Time (h)	Memory (MB)	Params
WESAD								
1	30	0.4404	0.2419	0.8110	0.8113	1.67	2461.20	203,080
2	60	0.5046	0.2434	0.7979	0.7970	1.69	2525.71	267,130
3	90	<u>0.5278</u>	0.2612	0.7877	0.7858	1.74	2598.04	331,180
4	120	0.5111	0.2658	0.7767	0.7787	1.74	2652.86	395,230
CogLoad								
1	30	0.0512	0.0287	0.9553	0.9521	1.35	1457.32	187,698
2	60	0.0568	0.0325	0.9496	0.9467	1.37	1713.54	235,448
3	90	<u>0.0723</u>	0.0363	0.9438	<u>0.9332</u>	1.45	1594.77	143,948
4	120	0.0695	0.0375	0.9427	0.9378	1.41	<u>1837.90</u>	281,198

both smaller and larger settings. While $d_m = 30$ yields a slight performance improvement on the CogLoad dataset (94.27% accuracy), the gain over $d_m = 18$ remains marginal when considering the associated increase in computational and memory costs.

A small hidden dimension of $d_m=9$ leads to underparameterization, limiting the model's ability to capture complex patterns and resulting in insufficient representation capacity. Conversely, increasing the hidden dimension substantially inflates the number of parameters and memory requirements without proportionally improving the representation of sensor input [26] [27]. The trade-off shows that a moderate setting, such as $d_m = 18$ on WESAD and $d_m = 30$ on CogLoad dataset, respectively, provides the best balance between accuracy and efficiency, making it ideal for resource-constrained wearable sensing applications.

4) *FFN Size Effects*: Table VII investigates the influence of the FFN dimension size on model performance. Configurations with FFN sizes of 30, 60, 90, and 120 are evaluated. The results indicate that smaller FFN sizes yield performance that is equal to or better than larger configurations. Specifically, on

the CogLoad dataset, an FFN size of 30 achieves the highest accuracy and F1 score while consuming the least memory. For WESAD, an FFN size of 30 also outperforms larger alternatives in terms of both accuracy and efficiency.

These findings suggest that large FFN layers may introduce overfitting and unnecessary computation. In Transformer architectures, FFN layers serve as position-wise transformations that expand and project features. In scenarios with relatively simple temporal structures, as seen in physiological data, excessive FFN capacity may learn noise instead of meaningful signal features. A smaller FFN not only helps in regularization but also reduces model latency and parameter count [28] [29].

B. Ablation Study

To further understand the impact of each architectural component, an ablation study was conducted, as shown in Table VIII. We analyze the combinations of local optimal configurations of the number of layers (L), attention heads (H), model dimension (d_m), and FFN size.

On WESAD, the combination of 1-layer and $d_m = 18$ ($L + d_m$) results in a remarkable accuracy of 88.41%, with only

TABLE VIII

ABLATION STUDY RESULTS ON WESAD AND COGLOAD DATASETS (LOCAL OPTIMAL CONFIGURATIONS: $c_layers = s_layers = 1$, $c_heads = s_heads = 1$, $d_m = 18/30$, $FFN = 30$); THE BEST RESULTS ARE SHOWN IN **bold**, AND THE WORST RESULTS ARE UNDERLINED)

Ablation Pair	L	H	d_m	FFN	Loss	MAE	Accuracy	F1	Train Time (h)	Memory Used (MB)	Params
WESAD											
Default	<u>5</u>	<u>3</u>	<u>30</u>	<u>120</u>	0.5111	0.2658	0.7767	0.7787	<u>1.74</u>	<u>2652.86</u>	<u>395,230</u>
L + H	1	1	30	120	0.4236	0.2135	0.8312	0.8235	0.39	408.92	81,910
L + d_m	1	3	18	120	0.2999	0.1486	0.8841	0.8815	0.42	575.20	30,874
L + FFN	1	3	30	30	0.3155	0.1407	0.8815	0.8819	0.43	634.03	43,480
H + d_m	5	1	18	120	0.4767	0.2567	0.7917	0.7898	1.31	1178.01	146,290
H + FFN	5	1	30	30	0.5020	0.2715	0.7957	0.7869	1.37	1368.12	203,080
d_m + FFN	5	3	18	30	0.3546	0.1649	0.8632	0.8644	1.59	2166.27	76,360
L + H + d_m	1	1	18	120	0.3855	0.2096	0.8446	0.8460	0.35	304.45	30,874
L + H + FFN	1	1	30	30	0.4021	0.2114	0.8299	0.8313	0.38	368.24	43,480
L + d_m + FFN	1	3	18	30	0.3859	0.2108	0.8479	0.8423	0.42	562.12	16,888
H + d_m + FFN	<u>5</u>	<u>1</u>	<u>18</u>	<u>30</u>	<u>0.6260</u>	<u>0.3090</u>	<u>0.7737</u>	<u>0.7566</u>	1.30	1058.63	76,360
L + H + d_m + FFN	1	1	18	30	0.4365	0.2289	0.8334	0.8235	0.37	287.53	16,888
CogLoad											
Default	<u>5</u>	<u>3</u>	<u>30</u>	<u>120</u>	0.4412	0.2145	0.8657	0.8681	<u>1.92</u>	<u>2798.77</u>	<u>281,198</u>
L + H	1	1	30	120	0.3711	0.1710	0.8920	0.8876	0.49	422.66	81,910
L + d_m	1	3	18	120	0.2634	0.1203	0.9245	0.9229	0.51	598.45	30,874
L + FFN	1	3	30	30	0.2402	0.1115	0.9261	0.9273	0.52	644.81	43,480
H + d_m	5	1	18	120	0.3980	0.1944	0.8788	0.8740	1.41	1213.22	146,290
H + FFN	5	1	30	30	0.4267	0.2022	0.8716	0.8665	1.48	1391.00	203,080
d_m + FFN	5	3	18	30	0.2915	0.1333	0.9130	0.9116	1.66	2234.20	76,360
L + H + d_m	1	1	18	120	0.3106	0.1581	0.9018	0.8993	0.46	318.90	30,874
L + H + FFN	1	1	30	30	0.3333	0.1634	0.8955	0.8920	0.48	377.81	43,480
L + d_m + FFN	1	3	18	30	0.3120	0.1558	0.9040	0.9025	0.52	574.33	16,888
H + d_m + FFN	<u>5</u>	<u>1</u>	<u>18</u>	<u>30</u>	<u>0.4925</u>	<u>0.2271</u>	<u>0.8571</u>	<u>0.8532</u>	1.39	1090.80	76,360
L + H + d_m + FFN	1	1	18	30	0.3548	0.1767	0.8874	0.8855	0.45	306.02	16,888

30K parameters—less than one-tenth of that of the full model. Additionally, the combination of 1-layer and $FFN = 30$ ($L + FFN$) also results in a remarkable accuracy of 88.15%, with only 43K parameters. Similarly, for CogLoad, ($L + d_m$) and ($L + FFN$) configurations achieve strong performance with minimal resource requirements. The best configuration ($L + FFN$) achieves 92.61% accuracy with significantly reduced training time and memory usage, almost similar but outperforming the accuracy of 92.45% in ($L + d_m$). Based on the ablation study, ($L + d_m$) reveals a higher contribution to performance across two datasets. On WESAD ($L + d_m$) parametrization yields the best result, outperforming ($L + FFN$) with the smallest number of 30K parameters; On CogLoad, the second best, but with a margin of only 0.173%.

These results highlight that shallow architectures, when combined with optimized low-dimensional embeddings and compact FFN layers, are sufficient for learning robust temporal and cross-modal representations. Furthermore, the minimal importance of the number of attention heads emphasizes that for low-resolution sensor inputs, even a single-head mechanism can capture modality-specific attention effectively. Across the ablation study of all locally-optimal configurations, our Efficient-Husformer demonstrates significant computational advantages. The most compact variant (e.g., $L = 1$, $H = 1$, $d_m = 18$, $FFN = 30$) achieves high classification accuracy of 83.34% and 88.74% on WESAD and CogLoad, respectively, with fewer than 20K parameters and less than 400 MB memory usage. These models can be trained within one hour and are suitable for real-time deployment on on-device or edge devices, such as wearables and smartphones.

C. Summary of Results and Analysis

Key Findings: 1) Firstly, shallower architectures (one cross-modal layer and one self-attention layer) consistently outperformed deeper alternatives on both WESAD and CogLoad datasets, achieving the highest accuracy and lowest error, while being the most computationally frugal. The deeper models suffered from over-parameterization and incurred further computational costs. 2) Secondly, configurations with the least number of attention heads (one cross-modal layer and one self-attention head) achieved the best balance of performance and efficiency. Adding additional attention heads to the configurations increased computational expenses without improving accuracy. 3) Finally, when model depth and either d_m or FFN were tuned jointly, optimal configurations were found. The best configuration was with one encoder layer and $d_m = 18$, offering the optimal balance of learning capacity and generalization. The patterns appearing in CogLoad were similar, but with a higher tolerance for model complexity, resulting in the best of $d_m = 30$, suggesting potentially higher separability of data.

The findings demonstrate that the Efficient-Husformer architecture significantly advances the balance of accuracy, efficiency, and resource cost trade-offs for multi-class stress classification from physiological data. In particular, the optimization strategy with decoupling of the cross-modal Transformer and self-attention Transformer was shown to be effective. It allows mode-fusion abilities and classification functionalities to be tuned independently, enabling more interpretability and resource efficiency.

VI. DISCUSSION AND FUTURE WORK

Progressive Transformer Scaling Across Diverse Datasets:

Although we propose a simple yet effective Efficient-Husformer model, achieving superior accuracy with a reduced memory footprint and training time, challenges regarding scalability and generalization in model design still remain. A potential direction is the adoption of multi-stage (progressive) model scaling for Transformer architectures across various domains, including Computer Vision [30] [31], Natural Language Processing [32] [33], and Time Series data [34] [35]. Developing such multi-stage (progressive) models to accommodate diverse datasets will be the focus of our future work.

VII. CONCLUSION

We introduced Efficient-Husformer, a Transformer-based architecture designed for multimodal physiological stress detection to emphasize efficiency, building upon the Husformer backbone. Efficient-Husformer separates cross-modal and self-attention mechanisms, enabling independent configuration of architectural components such as the number of layers (L), attention heads (H), model dimension (d_m), and feed-forward network size (FFN). This design supports fine-grained optimization for physiological computing tasks while reducing computational overhead.

Through extensive hyperparameter optimization (HPO) and ablation studies on two benchmark datasets: WESAD and CogLoad, we demonstrated that Efficient-Husformer outperforms the original Husformer while maintaining compact and deployable memory footprints, with the best configurations achieving accuracies of 88.41% and 92.61% (improvements of 13.83% and 6.98%, compared to the original Husformer) on WESAD and CogLoad datasets, respectively. Importantly, these gains were achieved with a dramatic reduction in model size and training time, highlighting that high-performance Transformer models can be made lightweight and resource-efficient through targeted architectural tuning.

REFERENCES

- [1] G. Keren and B. Schuller, “Convolutional rnn: An enhanced model for extracting features from sequential data,” *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 3412–3419, 2016.
- [2] B. Behniaein, A. Bhatti, D. Rodenburg, P. Hungler, and A. Etemad, “A transformer architecture for stress detection from ecg,” in *Proceedings of the 2021 ACM International Symposium on Wearable Computers, ISWC ’21*, (New York, NY, USA), p. 132–134, Association for Computing Machinery, 2021.
- [3] T. T. Vu, V. T. Huynh, and S. hyung Kim, “Multi-scale transformer-based network for emotion recognition from multi physiological signals,” in *Asian Conference on Pattern Recognition*, 2023.
- [4] L. Lange, B. Degenkolb, and E. Rahm, “Privacy-preserving stress detection using smartwatch health data,” in *GI-Jahrestagung*, 2023.
- [5] Z. Yang and H. Cao, “Decompose time and frequency dependencies: Multivariate time series physiological signal emotion recognition,” 2024.
- [6] J. Vazquez-Rodriguez, G. Lefebvre, J. Cumin, and J. L. Crowley, “Transformer-based self-supervised learning for emotion recognition,” 2022.
- [7] B. Klintron and A. Gracias, “Evaluating the performance of cnn, rnn, and transformer models for real-time activity recognition,” *Healthcare*, 12 2024.
- [8] A. Abu-Samah, D. Ghaffa, N. F. Abdullah, N. Kamal, R. Nordin, J. C. Dela Cruz, G. V. Magwili, and R. J. Mercado, “Deployment of tinyml-based stress classification using computational constrained health wearable,” *Electronics*, vol. 14, no. 4, 2025.
- [9] R. Wang, W. Jo, D. Zhao, W. Wang, A. Gupte, B. Yang, G. Chen, and B.-C. Min, “Husformer: A multimodal transformer for multimodal human state recognition,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 16, no. 4, pp. 1374–1390, 2024.
- [10] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven, “Introducing wesad, a multimodal dataset for wearable stress and affect detection,” in *Proceedings of the 20th ACM international conference on multimodal interaction*, pp. 400–408, 2018.
- [11] M. Gjoreski, T. Kolenik, T. Knez, M. Luštrek, M. Gams, H. Gjoreski, and V. Pejović, “Datasets for cognitive load inference using wearable sensors and psychological traits,” *Applied Sciences*, vol. 10, no. 11, 2020.
- [12] K. T. Chitty-Venkata, M. Emani, V. Vishwanath, and A. Soman, “Neural architecture search for transformers: A survey,” *IEEE Access*, vol. PP, pp. 1–1, 10 2022.
- [13] S. A. Hosseini Aqajari, E. K. Naeini, M. A. Mehrabadi, S. Labbaf, N. Dutt, and A. M. Rahmani, “pyeda: An open-source python toolkit for pre-processing and feature extraction of electrodermal activity,” *Procedia Computer Science*, vol. 184, pp. 99–106, 2021. The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.
- [14] P. Bobade and V. M., “Stress detection with machine learning and deep learning using multimodal physiological data,” *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, pp. 51–57, 2020.
- [15] Y. Su, L. Ge, and G. Wei, “Random forest model predicts stress level in a sample of 18,403 college students,” in *Proceedings of the 2024 4th International Conference on Artificial Intelligence, Big Data and Algorithms, CAIBDA ’24*, (New York, NY, USA), p. 588–593, Association for Computing Machinery, 2024.
- [16] Y. Yao, M. Papakostas, M. Burzo, M. Abouelenien, and R. Mihalcea, “Muser: Multimodal stress detection using emotion recognition as an auxiliary task,” *arXiv preprint arXiv:2105.08146*, 2021.
- [17] S. Ziaratnia, T. Laohakangvalvit, M. Sugaya, and P. Sripijan, “Multimodal deep learning for remote stress estimation using cct-lstm,” in *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 8321–8329, 2024.
- [18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *NIPS’17*, (Red Hook, NY, USA), p. 6000–6010, Curran Associates Inc., 2017.
- [19] D. Theodoropoulos, F. Stahl, and M. Lindauer, *Hyperparameter Importance Analysis for Multi-Objective AutoML*. IOS Press, Oct. 2024.
- [20] E. Haapalainen, S. Kim, J. F. Forlizzi, and A. K. Dey, “Psychophysiological measures for assessing cognitive load,” in *Proceedings of the 12th ACM International Conference on Ubiquitous Computing, UbiComp ’10*, (New York, NY, USA), p. 301–310, Association for Computing Machinery, 2010.
- [21] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” *arXiv preprint arXiv:1611.03530*, 2016.
- [22] L. Brigato and L. Iocchi, “A close look at deep learning with small data,” in *2020 25th international conference on pattern recognition (ICPR)*, pp. 2490–2497, IEEE, 2021.
- [23] E. Voita, D. Talbot, F. Moiseev, R. Senenrich, and I. Titov, “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5797–5808, 2019.
- [24] S. Bhojanapalli, C. Yun, A. S. Rawat, S. J. Reddi, and S. Kumar, “Low-rank bottleneck in multi-head attention models,” in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020. arXiv preprint.
- [25] T. Hrycej, B. Bermeitinger, and S. Handschuh, “Number of attention heads versus number of transformer-encoders in computer vision,” in *13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KDIR)*, 2022. arXiv preprint.
- [26] S. Yao, Y. Hu, Y. Zhao, S. Zhang, and T. Abdelzaher, “Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework,” *arXiv preprint arXiv:1705.07356*, 2017.
- [27] J. Tang, F. Wu, F. Zhang, and Y. Pan, “Layer-wise training convolutional neural networks with smaller filters for human activity recognition using wearable sensors,” in *Proceedings of the 2020 International Symposium on Wearable Computers*, pp. 60–64, ACM, 2020.
- [28] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the big data paradigm with compact transformers,” *arXiv preprint arXiv:2104.05704*, 2021.

- [29] T. Pessoa Pires, A. V. Lopes, Y. Assogba, and H. Setiawan, “One wide feedforward is all you need,” *arXiv preprint arXiv:2309.01826*, 2023.
- [30] C. Li, B. Zhuang, G. Wang, X. Liang, X. Chang, and Y. Yang, “Automated progressive learning for efficient training of vision transformers,” *arXiv preprint arXiv:2203.14509*, 2022.
- [31] C. Li, J. Zhang, S. Lin, Z. Yang, J. Liang, X. Liang, and X. Chang, “Efficient training of large vision models via advanced automated progressive learning,” *arXiv preprint arXiv:2410.00350*, 2024.
- [32] X. Gu, L. Liu, H. Yu, J. Li, C. Chen, and J. Han, “On the transformer growth for progressive bert training,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 5174–5180, 2021.
- [33] S. Shen, P. Walsh, K. Keutzer, J. Dodge, M. Peters, and I. Beltagy, “Staged training for transformer language models.” *arXiv:2203.06211*, 2022.
- [34] W. Zhou, H. Wang, Y. Zhang, C. Long, Y. Yang, and D. Wang, “Multi-scale progressive gated transformer for physiological signal classification,” in *Asian Conference on Machine Learning*, pp. 1293–1308, PMLR, 2023.
- [35] D. Liang, H. Zhang, D. Yuan, B. Zhang, and M. Zhang, “Minusformer: Improving time series forecasting by progressively learning residuals,” *arXiv preprint arXiv:2402.02332*, 2024.