Lightweight Prediction Method for Blade Aerodynamic Performance Based on CNN-Transformer

Authors: Zheng Jiang, Zhang Xiaojiu, Xu Xiaohan, Cao Huiping, Qingfei Lu, Pengcheng Zhao, Lu Qingfei

Date: 2025-05-14T17:32:35+00:00

Abstract

Accurate prediction of compressor blade aerodynamic performance is crucial for improving aero-engine design efficiency and reducing energy consumption in the turbomachinery industry. Currently, both the high computational cost of traditional simulations and deep learning methods represented by standard Convolutiona1Neural Networks (CNN) and hybrid architectures integrating Squeeze-and-Excitation (SE) modules with ConvolutionalBlockAttentionModule (CBAM) still encounter trade-off challenges between prediction accuracy and computational efficiency. This paper proposes a CNN neural network architecture (CNN-Transformer) that integrates Transformer multi-head attention mechanism. Compared with traditional convolutional neural networks, CNN-Transformer significantly improves prediction performance metrics while maintaining computational efficiency—reducing Root Mean Square Error (RMSE) by 40.19% (to 0.0064), Mean Absolute Percentage Error (MAPE) by 31.94% (to 0.98%), and accompanied by a 27.60% decrease in EstimatedTotalSize. Experiments demonstrate that this architecture can effectively establish a nonlinear mapping relationship between critical inlet parameters (Mach number, Reynolds number, inlet angle, etc.) and outlet aerodynamic characteristics, surpassing existing models in both prediction accuracy and inference speed. This method offers an efficient computational solution for real-time evaluation of blade performance during aerodynamic design iteration processes, substantially accelerating the design optimization workflow while meeting engineering accuracy requirements.

Full Text

Lightweight Prediction Method for Blade Aerodynamic Performance Based on CNN-Transformer

Jiang Zheng[1], Xiaojiu Zhang[2], Xiaohan Xu[2], Huiping Cao[1], Qingfei Lu[1,3]*, Pengcheng Zhao[4]
[1] School of Aerospace, Xihua University; [2] China Academy of Launch Vehicle Technology; [3] Tianfu Jiangxi Laboratory; [4] AECC Sichuan Gas Turbine Establishment

Abstract

Accurate prediction of compressor blade aerodynamic performance is crucial for enhancing aero-engine design efficiency and reducing energy consumption in turbomachinery industries. Current deep learning methodologies, represented by standard Convolutional Neural Networks (CNN) and hybrid architectures integrating Squeeze-and-Excitation (SE) and Convolutional Block Attention Module (CBAM), face challenges in balancing prediction accuracy with computational efficiency. This study proposes a hybrid CNN-Transformer neural network architecture that innovatively incorporates the Transformer's multi-head attention mechanism into a CNN framework. Compared with traditional Convolutional Neural Networks, CNN-Transformer achieves superior performance metrics while maintaining computational efficiency—reducing Root Mean Squared Error (RMSE) by 40.19% (to 0.0064) and Mean Absolute Percentage Error (MAPE) by 31.94% (to 0.98%), accompanied by a 27.60% reduction in Estimated Total Size. Rigorous validation demonstrates that the proposed architecture effectively establishes nonlinear mappings between critical inlet parameters (Mach number, Reynolds number, flow angle) and outlet aerodynamic characteristics, outperforming existing models in both prediction accuracy and inference speed. This advancement delivers an efficient digital solution for real-time compressor blade performance evaluation during aerodynamic design iterations, significantly accelerating the design optimization process while maintaining engineering precision requirements.

Keywords: blade profiles; CNN-Transformer; convolutional neural networks; aerodynamic prediction; multi-head attention mechanism

Introduction

Faced with the growing complexity of aerospace systems and the heightened performance demands of next-generation fighter aircraft, propulsion system design is undergoing a paradigm shift toward higher integration and efficiency. As the technical core of such systems, modern aero-engines must meet stringent performance metrics, prompting the need for more refined and efficient aerodynamic analysis methods. Against this backdrop, the accurate and rapid prediction of compressor blade aerodynamic performance has become a fundamental capa-

bility that directly impacts propulsion system efficiency and guides the design iteration process [**?**].

As a critical subsystem of aero-engines, the compressor plays a pivotal role in determining overall engine performance, with its aerodynamic characteristics profoundly influencing design outcomes. Accurate determination or prediction of these parameters represents a fundamental prerequisite for engine performance optimization. To enhance compressor efficiency and operational performance, research on blade aerodynamic performance has emerged as a key research direction. Accurate prediction of compressor blade performance not only improves propulsion system effectiveness but also transforms engine design methodologies. Currently, mainstream approaches include experimental validation and Computational Fluid Dynamics (CFD) simulation [**?**]. While these methods offer clear advantages in prediction accuracy, their high consumption of experimental resources and lengthy simulation cycles still limit their efficiency in engineering applications. Consequently, developing prediction tools with fast response times and high cost-effectiveness to meet modern engineering demands for accelerated design iteration and performance evaluation has become an urgent priority.

In recent years, deep learning methods have achieved transformative advances in computer vision and computational science. By extracting high-dimensional nonlinear relationships and uncovering underlying physical laws, these techniques demonstrate significant potential for aerospace engineering applications. In turbomachinery research, deep learning has shown powerful capabilities in prediction, fault detection, and optimization. Wang et al. [**?**] proposed a dual-path Convolutional Neural Network (Dual-CNN) architecture capable of reconstructing flow fields within error ranges suitable for aerodynamic optimization frameworks, effectively revealing performance variation mechanisms. Meanwhile, Ma et al. [**?**] combined a U-Net for three-dimensional flow field reconstruction with a 1D-CNN for one-dimensional performance parameter prediction, achieving mean relative errors of less than 1% and 2% for flow field reconstruction and performance parameter estimation, respectively. Furthermore, Wang et al. [**?**] developed an integrated method using multi-scale CNN and Support Vector Machine (SVM) classifiers, achieving over 99% accuracy in identifying compressor stall modes across different rotational speed ranges. Additionally, Li et al. [**?**] significantly enhanced surge prediction capability through a Northern Goshawk-optimized NGO-LSTM network, enabling reliable 1-second advance warning for both pulse-type and modal-type surge events, providing a critical time window for active clearance control system response. Fan et al. [**?**] employed reinforcement learning algorithms to discover optimal strategies, effectively expanding the Pareto frontier, with optimal designs achieving a 13.1% stall margin improvement without impacting peak efficiency—surpassing results obtained using Kriging models and NSGA-II algorithms. Liu et al. [**?**] integrated CNN-PINN-DRL for airfoil optimization, significantly improving lift-to-drag ratio. Wang et al. [**?**] combined CNN with the NSGA-II algorithm to optimize blades, obtaining blade geometry optimization results in short timeframes, achieving maximum power coefficients and broadest high-efficiency operating

ranges.

Throughout these research efforts, the predictive performance of CNNs has played a crucial role. Beyond the need for pure performance prediction, optimization workflows also require a method to rapidly obtain accurate solutions corresponding to the Pareto frontier. Therefore, this study establishes a deep learning-based framework for rapid prediction of blade aerodynamic performance, achieving high-precision prediction by mapping inlet boundary conditions to outlet flow parameters. While improving prediction accuracy, this work specifically focuses on optimizing the computational efficiency of model structures, aiming to achieve intelligent inference capabilities deployable in edge computing scenarios for aerospace systems.

Contributions

This paper's contributions are specifically:

1. **Introduction of CNN-Transformer Hybrid Architecture**: This study introduces a hybrid architecture that fuses the multi-head attention mechanism from Transformers into Convolutional Neural Networks (CNN), aiming to simultaneously achieve local feature extraction and global dependency modeling. This framework effectively overcomes the limitations of traditional single-structure models in capturing multi-scale aerodynamic interactions in compressor blade flows, significantly enhancing feature representation capabilities and cross-scale generalization performance.
2. **Establishment of Efficient Compressor Blade Prediction Evaluation Framework**: Diverse evaluations were conducted on the same aerodynamic dataset. The results demonstrate that the proposed CNN-Transformer model achieves a 31.94% reduction in Mean Absolute Percentage Error (MAPE) compared to standard CNN architecture. Simultaneously, prediction time is dramatically reduced from several hours required by traditional CFD solvers to under one second, providing real-time feedback support for multi-objective compressor blade optimization design.

Methodology

Data Preparation

In this study, we constructed a subsonic dataset comprising 1,980 image samples, with each sample corresponding to specific boundary conditions for a compressor blade profile, including inlet Mach number, Reynolds number, and inlet angle, annotated with high-fidelity aerodynamic parameters obtained from NUMECA simulations, such as outlet Mach number. The outlet Mach number was designated as the target variable for model learning. To efficiently generate simulation data, IGG automation scripts were employed for batch modeling, with Turbo enabling batch computation. To reduce computational overhead, this

study adopted a two-dimensional modeling strategy, which supports dataset scaling while maintaining acceptable accuracy.

Data Processing:

1. **Image Normalization**: Input image pixel intensity values were linearly scaled from the original [0, 255] range to [0, 1]. This normalization operation enhances numerical stability during training, prevents issues like gradient vanishing or explosion, and reduces model sensitivity to input amplitude variations.
2. **Label Normalization**: The target outlet Mach numbers were similarly normalized. This step helps smooth regression outputs, reducing model dependence on absolute label values and thereby improving prediction accuracy and generalization capability.

Model Structure Design

To enhance training stability and further suppress overfitting, Batch Normalization operations were introduced after each convolutional layer to mitigate internal covariate shift and accelerate model convergence. Additionally, max pooling layers were added after each convolutional layer to downsample feature maps, reducing spatial dimensions and thereby decreasing computational complexity while enhancing model robustness to input variations.

Model Selection

To prevent model overfitting, this study implemented measures across three aspects: image preprocessing, data processing, and model structure design, as detailed below.

Image Preprocessing: To enhance training dataset diversity and improve model generalization capability, various data augmentation techniques were applied, including geometric transformations (such as rotation, scaling) and controlled color perturbations. These augmentation operations aim to simulate variations caused by aerodynamic boundary condition fluctuations (such as inlet angle, Mach number, Reynolds number), thereby embedding upstream flow characteristic information into blade geometry images. Through this approach, the model learns more robust and transferable mappings between geometric features and aerodynamic performance, adapting to broader operating condition variations. Additionally, increased input data diversity effectively reduces overfitting risk.

Convolutional Neural Networks (CNN) are a neural network architecture specifically optimized for spatial data processing, particularly adept at extracting hierarchical feature patterns from geometrically structured inputs. Its basic architecture primarily implements functionality through three core mechanisms: (1) local feature extraction using convolutional kernels (filters), (2) feature map dimensionality reduction through pooling operations, and (3) nonlinear decision

boundary formation via fully connected layers [**?**]. Building upon this CNN foundation, this study further developed two enhanced architectures (SS-CNN and CNN-T) through modular extensions to improve performance. To meet embedded deployment requirements, we conducted systematic network structure exploration based on the dataset, evaluating various candidate network configurations according to computational efficiency and prediction accuracy metrics (see Table 1).

Table 1 Performance Metrics for Different Layers

Through comprehensive performance-computational overhead trade-off analysis, the four-layer Convolutional Neural Network (CNN) structure (see Figure 1) achieved optimal balance between operational efficiency and predictive capability. The selected architecture employs a 3$×$3 convolutional kernel design, which significantly reduces computational load compared to traditional large kernel schemes while maintaining performance:

1. **3$×$3 Convolutional Kernel Configuration**: 3.4931 GFLOPs
2. **7$×$7 Convolutional Kernel Configuration**: 17.6508 GFLOPs

This small kernel strategy reduces computational complexity by 80.22% while preserving the ability to identify critical features such as compressor blade profiles.

Figure 1 [Figure 1: see original paper] CNN

CNNs effectively capture local detail information in images, such as edges, textures, and shape features, through their local perception mechanism and hierarchical feature extraction process. Through layer-by-layer convolutional operations, CNNs progressively abstract image content across different scales, thereby preserving and expressing micro-structures and local variation features. However, due to CNN's inherently limited receptive field, its ability to model global dependency relationships is weak. Additionally, strong sequential dependencies between CNN layers limit parallel computational efficiency, consequently restricting model scalability in large-scale application scenarios.

SS-CNN (SE-SAM-CNN) integrates SAM and SE modules into the CNN architecture. The Spatial Attention Module (SAM) is a key component of the Convolutional Block Attention Module (CBAM) architecture. CBAM serves as an efficient attention mechanism that collaboratively enhances feature discriminability across both channel and spatial dimensions, achieving targeted feature strengthening while maintaining structural consistency of learned representations.

CBAM's architectural advantage primarily stems from its sequentially integrated two complementary sub-modules: (1) Channel Attention Module (CAM), which dynamically adjusts feature responses of each channel through adaptive weighting mechanisms, and (2) Spatial Attention Module (SAM), which highlights feature expression of prominent spatial regions through contextual feature association mechanisms. Experimental evidence indicates that the

sequential channel-first approach (CAM→SAM) yields superior performance compared to parallel or reversed sequential strategies [**?**].

Figure 2 [Figure 2: see original paper] CBAM-CNN

The core of CBAM's unique capability lies in its dual pooling strategy: max pooling and average pooling. These complementary pooling mechanisms achieve critical performance improvements over traditional single pooling convolution operations.

The Channel Attention Module (CAM, see Figure 3) extracts channel-dimensional features from feature map F ^{C×H×W} processed by CNN convolution and pooling, obtaining feature representations through average pooling and max pooling. The extracted features are fed into a Multi-Layer Perceptron (MLP) for feature aggregation and concatenation, generating channel attention map M_c(F), which is then element-wise multiplied ( ) to obtain enhanced feature map F . The calculation formula is as follows (where σ represents the Sigmoid activation function):

M_c(F) = σ(MLP([AvgPool(F); MaxPool(F)])) (1)
F = M_c(F) F

Figure 3 [Figure 3: see original paper] CAM

The Spatial Attention Module (SAM, see Figure 4) further processes the output feature map F from CAM. After performing average pooling and max pooling on the input feature map, SAM generates the final enhanced feature map F through convolution operations. The calculation formula is as follows (where f_{n×n} represents convolution operations using n×n kernels):

M_s(F) = σ(f_{n×n}([AvgPool(F); MaxPool(F)])) (3)
F = M_s(F) F

Figure 4 [Figure 4: see original paper] SAM

However, in our experimental testing, due to the relatively simple dataset and the high computational complexity of CBAM modules, fitting effects were sub-optimal while computational overhead remained significant (see Table 2). Here, CBAM_{x_y} denotes adding x CBAM modules to the network and placing them at y positions. Therefore, we optimized CBAM by first reducing the complexity of the Channel Attention Module (CAM) and replacing it with the SE (Squeeze-and-Excitation) module. The SE module consists of two core steps: Squeeze and Excitation [**?**].

Compared with the Spatial Attention Module (SAM), Squeeze uses only Global Average Pooling (GAP), which averages values across all spatial positions in each channel to obtain representative features for each channel. This method compresses global information into channel descriptors. It transforms the two-dimensional feature map F (H × W) into a one-dimensional channel descriptor z_c, significantly reducing computational complexity. Simultaneously, by calculating the average across the entire feature map, Squeeze provides a global

representation, enabling channel features to represent global information rather than being limited to local regions. The calculation formula is as follows [**?**]:

z_c = G_{sq}(F_c) = (1/(H×W)) Σ_i Σ_j F_c(i, j)
- F_c(i, j) represents the value at a specific position on channel c
- G denotes the function computation

The Excitation step uses a simple gating mechanism combined with a Sigmoid activation function. To limit model complexity and enhance generalization capability, the gating mechanism is implemented by constructing two fully connected layers (FC) combined with nonlinear activation functions (ReLU). The formula is as follows:

s = σ(W_2 δ(W_1 z))
- δ denotes the ReLU activation function
- W_1 and W_2 are parameter matrices

The final output $\tilde{x}$_c is obtained by element-wise multiplication of each channel's feature map matrix F_c with the channel weight scalar s_c, performing scaling for each channel:

$\tilde{x}c$ = G{scale}(F_c, s_c) = s_c · F_c

Table 2 Comparison of Metrics Across Different CBAM-CNN Architectures

Through analysis of formulas for both Channel Attention Module (CAM) and Squeeze-and-Excitation (SE) module, we can see that the SE module has relatively lower computational overhead. However, we still performed further optimization. Since CNNs are inherently biased toward capturing local features and already emphasize prominent regions through max pooling, the role of Spatial Attention Module (SAM) is primarily to evaluate the importance of local regions within the spatial domain—functionally overlapping to some extent with max pooling effects.

Figure 5 [Figure 5: see original paper] SS-CNN

Therefore, to minimize computational overhead, we chose to integrate SAM modules only in the final network layer. Consistent with the sequential structure of Convolutional Block Attention Module (CBAM), we connected SE and SAM modules sequentially, forming what we term the SS module. This design ultimately led to the SE-SAM-CNN model (Figure 5). Experimental results on our dataset demonstrated excellent performance of the SE-SAM-CNN model, with detailed results presented in subsequent sections.

CNN-Transformer is a hybrid model combining Convolutional Neural Networks (CNN) and Transformer, aiming to leverage CNN's local perception capability and Transformer's global modeling capacity.

Transformers have received widespread attention in the vision domain due to their excellent performance in natural language processing [**?**]. Dosovitskiy et al. [**?**] demonstrated that the model also performs exceptionally well in vision

tasks. Transformer is based on self-attention mechanisms, adopting an encoder-decoder structure built by stacking self-attention layers and fully connected layers. Unlike traditional Recurrent Neural Networks (RNN) and CNN, Transformer completely abandons recursion and convolution, relying on self-attention mechanisms to capture global dependencies while providing stronger parallelism and significantly reducing training time [**?**].

CNN's output is a 4D feature map in format (B, C, H, W), where B represents batch size, C denotes channel number, and H and W represent feature map height and width respectively. Since Transformer processes data in sequential format, CNN output feature maps need to be flattened into sequences before introducing Transformer, such as (H×W, B, C). This flattened feature map can serve as input embeddings for Transformer processing.

Subsequently, multi-head attention mechanisms model global relationships of input features. Unlike single attention heads that focus on only one subspace per computation and cannot comprehensively learn input sequence diversity, multi-head attention mechanisms learn different dependencies within input sequences across multiple subspaces through multiple parallel attention heads.

Each attention head independently operates on the input sequence, allowing the model to capture information from multiple perspectives. Query (Q), Key (K), and Value (V) matrices of the input sequence are linearly projected into multiple low-dimensional subspaces (each with dimension d_k or d_v). Attention is computed independently within each subspace, after which results from all heads are concatenated and linearly transformed to obtain final output (Figure 6, left). The calculation formula is as follows [**?**]:

MultiHead(Q,K,V) = Concat(head_1, …, head_n)W^O

Each head's calculation formula is:
head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) (9)

- W^O ^{hd_v×d_{model}} is the concatenated output
- Q ^{d_{model}×d_k}, K ^{d_{model}×d_k}, V ^{d_{model}×d_v}
- W_i^Q, W_i^K, W_i^V are learnable linear projection matrices for mapping Q, K, V to different subspaces
- d_{model} is the total dimension of input features; d_k is the feature dimension per head

Each attention head is computed in parallel. After computation, results from all heads are concatenated to obtain output with dimension h d_v. Finally, the output is projected back to d_{model} dimension through the W^O matrix.

In our model, the fourth layer's output feature map has shape (50, 128, 8, 8). By flattening the feature map into a sequence of (64, 50, 128), where each sequence has length 64 and feature dimension d_{model} = 128, we applied a four-head

self-attention mechanism (with each head having dimension d_k = 32). Using multi-head self-attention, we captured global relationships. The output was stabilized through residual connections and layer normalization. Finally, the feature map shape was restored by reshaping the sequence back to (50, 128, 8, 8).

We chose to add self-attention modules only between the fourth convolutional layer and fully connected layer (Figure 6, right). Compared with adding multi-head attention modules after every layer or placing them in lower layers (such as layers 1 and 2), this approach reduces computational overhead and complexity. For instance, adding it to the first layer would require 12.50 GiB of GPU memory, which is unacceptable. In CNN convolution operations, early layers primarily capture low-level features (such as edges and textures), while attention mechanisms capture global dependencies. Placing attention modules in higher layers (i.e., after Conv4) enables more effective utilization of advanced features extracted by CNN while modeling global relationships. This approach aligns with our design objective of ensuring accuracy while achieving low complexity.

Figure 6 [Figure 6: see original paper] Multi-Head Attention (left); CNN-Transformer (right)

Experimental Environment and Platform

Experimental Environment: PyTorch 1.11.0; Python 3.8 (Ubuntu 20.04); CUDA 11.3
Platform Hardware: CPU: 18 vCPU AMD EPYC 9754 128-Core Processor, 128 cores, 256 threads; GPU: RTX 4090D (24GB) × 1

Evaluation Metrics

Evaluation metrics primarily include two aspects: accuracy and computational cost.

Accuracy is primarily reflected through Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE).

- **MAPE**: A commonly used metric for measuring model prediction error, particularly suitable for regression tasks. It represents the average percentage error between predicted and true values, intuitively reflecting model prediction precision.
- **RMSE**: A commonly used metric for measuring error between predicted and true values, particularly suitable for regression tasks. It reflects the average magnitude of errors between predictions and actual values. Smaller RMSE indicates closer alignment between predictions and true values.

Mathematical formulas are as follows:

$$\text{MAPE} = (1/n)\ \Sigma_i\ |(y_i - \hat{y}_i)/y_i| \times 100\% \quad (10)$$
$$\text{RMSE} = \sqrt{(\ (1/n)\ \Sigma_i\ (y_i - \hat{y}_i)^2\ )}$$

- y_i: true value
- ŷ_i: predicted value
- n: number of samples

Computational cost primarily includes: Floating Point Operations Per Second (FLOPs), total number of trainable parameters (Params), training time per epoch (Time), and Estimated Total Size (a metric for storage space).

- **FLOPs**: In deep learning, FLOPs is the standard metric for measuring computational complexity required for one forward propagation. Total floating-point operations directly reflect algorithmic complexity. FLOPs is hardware-independent; smaller values indicate higher computational efficiency and faster model inference speed. It is an important metric for designing low-complexity models.
- **Params**: Represents parameters that need updating through backpropagation during model training, serving as a key indicator of model size and storage requirements. Params determines model storage size (total bytes of parameters) and directly affects operational efficiency on hardware devices. Params measures static model size (number of parameters requiring storage), while FLOPs measures dynamic computational complexity (computations required for one forward pass).

FLOPs calculations primarily involve various layers (convolutional, pooling, fully connected, etc.):

- Convolutional layer: FLOPs = 2 ·H_{out} ·W_{out} ·C_{in} ·K_h ·K_w ·C_{out}
- Pooling layer: FLOPs = H_{out} ·W_{out} ·C_{out} ·M_h ·M_w
- Fully connected layer: FLOPs = 2 ·N_{in} ·N_{out}
- H_{out}, W_{out}: output feature map height and width
- C_{in}, C_{out}: input and output feature map channels
- K_h, K_w: convolutional kernel height and width
- M_h, M_w: pooling window height and width
- N_{in}, N_{out}: number of input and output nodes

Time: Helps monitor training progress, identify performance bottlenecks, compare efficiency across different models and hyperparameter configurations, and provide comprehensive understanding of training processes.

Estimated Total Size: Typically used to represent expected total data volume or resources during data loading and model training. It helps predict required resources, ensures sufficient resources for program execution, and enables further optimization through model and resource adjustments.

Experimental Results Analysis

We systematically evaluated the prediction performance of standard CNN, SS-CNN, and CNN-Transformer (CNN-T) architectures across progressively increasing training dataset sizes, with results summarized in Table 3. For each data size setting, training subsets were randomly sampled from the complete dataset to ensure statistical consistency and mitigate sampling bias. Results reveal CNN-T's robustness and generalization capability across different data availability scenarios:

1. Small Dataset Stage (10%-20% of full dataset, 198-396 samples)

In this stage, CNN-T achieved the lowest MAPE (1.55% and 6.33%), significantly outperforming CNN (2.18% and 17.84%) and SS-CNN (1.93% and 15.51%). This highlights CNN-T's superior feature extraction capability under data scarcity, attributable to Transformer's self-attention mechanism that can focus on critical spatial dependencies even with limited training samples.

2. Medium Dataset Stage (30%-50% of full dataset, 594-990 samples)

Within this range, standard CNN demonstrated the most stable performance, with MAPE decreasing from 40.72% to 9.66%, outperforming both SS-CNN and CNN-T. CNN's simpler architecture with fewer parameters enables effective generalization and avoids overfitting. In contrast, CNN-T's performance lagged because its attention mechanisms had not yet been fully utilized, requiring more data to effectively learn global dependencies. SS-CNN's accuracy also fluctuated, suggesting its hybrid attention structure may not have been fully optimized at this scale.

3. Transition to Large Dataset Stage (60%-90% of full dataset, 1188-1782 samples)

In this stage, SS-CNN emerged as the most effective model, achieving lower MAPE than CNN and CNN-T in most cases. For instance, with 1782 samples, SS-CNN achieved MAPE of 2.20%, outperforming CNN (2.77%) and CNN-T (2.47%). Its ability to simultaneously model channel and local spatial features enables more efficient feature refinement. CNN's performance improvement plateaued, while CNN-T continued improving but had not yet fully activated its global modeling capability.

4. Full Data Utilization (100%, 1980 samples)

Figure 7 [Figure 7: see original paper] RMSE
Figure 8 [Figure 8: see original paper] MAPE

With the full dataset, CNN-T achieved its best performance, attaining the lowest MAPE (0.98%) and RMSE (0.0064), significantly outperforming CNN (MAPE: 1.44%) and SS-CNN (1.36%). At this scale, the 4-head self-attention mechanism

in CNN-T was fully trained, enabling effective capture of long-range dependencies and global flow field features. Its hybrid design successfully combined local and global modeling, surpassing the capability limitations of standard CNN.

Table 4 Comparison of Metrics Across Different Architectures (Data Size: 1980)

Summary Comparison:

1) CNN-T demonstrates near-maximal error reduction with increasing data volume, particularly showing continuous performance improvement during transition from small to large datasets, ultimately achieving MAPE below 1%.

2) SS-CNN exhibits good cost-effectiveness balance on medium-to-large datasets, benefiting from its dual attention mechanism.

3) Standard CNN remains the optimal choice for rapid deployment with only medium-scale data due to its low complexity and stable generalization capability.

These findings confirm architecture-dependent performance scaling laws in aerodynamic prediction tasks. Model selection should be guided by available data volume, required accuracy, and computational constraints, with CNN-T recommended for scenarios with sufficient data and high accuracy requirements.

Overall, CNN-T demonstrates the most stable performance fluctuations and superior generalization capability across different data scales. Even under extremely limited data conditions, it effectively captures global dependencies, achieving the lowest prediction error at full data scale, fully demonstrating advantages of its global modeling architecture. Although CNN-T's error metrics were slightly higher than CNN in the 50%-70% data range, its performance rapidly stabilized and ultimately surpassed other models with increasing data volume, highlighting its robustness and adaptability to complex, nonlinear data distributions.

Under full-data training conditions, CNN-T significantly outperformed CNN and SS-CNN in key error metrics, reducing RMSE by 40.19% and MAPE by 31.94%, fully meeting our aerodynamic prediction accuracy requirements. Regarding computational overhead (measured in GFLOPs), SS-CNN and CNN-T showed almost no increase compared to standard CNN. However, SS-CNN's training time per epoch increased by approximately 50%, while CNN-T only increased by 27.10%. This indicates that Transformer module incorporation has minimal impact on training efficiency, providing a good compromise between performance and computational cost.

Furthermore, CNN-T achieved a 27.60% reduction in estimated model size, while SS-CNN increased by 3.51%. This reduction not only enhances deployability in resource-constrained environments but also further strengthens CNN-T's applicability in real-time or embedded applications with limited storage and

memory.

Figure 8 [Figure 8: see original paper] Deviation between predicted and true values
Figure 9 [Figure 9: see original paper] Impact point of predicted value within 1% error range of true value

In summary, although SS-CNN is competitive in some metrics, CNN-T consistently provides better overall balance among prediction accuracy, computational efficiency, and storage cost, reflecting our design objectives of optimizing both performance and practicality. Compared to these methods, Niu et al. [**?**] achieved less than 2% mean squared error when predicting compressor blade aerodynamic loads using deep learning models. Wu et al. [**?**] combined convolutional neural networks with fully connected layers to predict airfoil aerodynamic parameters, achieving relative errors below 1.5% and 1.2% for lift-to-drag ratio coefficient and pressure distribution, respectively. Our proposed CNN-T model demonstrates highly competitive performance, even surpassing these models in some cases.

Further Exploration: Can Module Fusion Surpass CNN-Transformer for Better Low-Complexity Blade Prediction?

The above experimental results demonstrate that the CNN-Transformer (CNN-T) framework not only surpasses standard CNN and SS-CNN in prediction accuracy but also maintains high computational efficiency. These results highlight CNN-T's robustness in capturing both local and global flow features, which is crucial for aerodynamic performance prediction of blade geometries. However, the significant improvement brought by SS modules to CNN raises an important question in our low-complexity prediction model research:

Can further fusion of SS modules with Transformer architecture surpass CNN-T in both accuracy and computational cost, thereby better serving our low-complexity design objectives for blade performance prediction models?

To this end, we designed two fusion-based architectures:

1) **SS-CNN-T**: Adds a Multi-Head Attention (MHA) mechanism at the end of SS modules, representing a direct module stacking strategy.

2) **SAM-CNN-T**: Embeds the Spatial Attention Module (SAM) within MHA, representing a more integrated and logical design.

The first approach prioritizes a "local-first, global-second" modeling strategy. Specifically, we embed SAM before MHA, first emphasizing key spatial regions, thereby enabling Transformer to focus global modeling on the most important areas. This design also aligns with the overall objective of reducing unnecessary computation, maintaining model simplicity, and improving inference efficiency—critical for low-complexity prediction.

The second approach: Embedding MHA within SAM could lead to processing inefficiency, as global dependencies are extracted from original feature maps while subsequent SAM operations may dilute refined outputs. This not only increases computational cost but also weakens interpretability of local-global fusion, particularly in space-focused tasks.

However, experimental results show that neither SS-CNN-T nor SAM-CNN-T architectures could surpass the original CNN-T. In fact, both designs added computational complexity without significantly improving accuracy—an effect we interpret as negative optimization. A plausible explanation lies in our dataset's nature: blade performance data originates from two-dimensional profiles with clear boundaries and relatively simple spatial structures, which CNN-T's hybrid local-global feature extraction already handles well. Further addition of local attention modules may cause redundancy or even interfere with global feature learning.

Additionally, our use of small convolutional kernels—designed to minimize model size and GFLOPs—reduces effectiveness of channel and spatial attention mechanisms, as these are typically designed to complement larger receptive fields. In this low-complexity configuration, Transformer's multi-head attention mechanism provides the most computationally efficient performance improvement, echoing the principle that simplicity is effective.

These findings reaffirm that CNN-T remains the optimal choice for low-complexity aerodynamic performance prediction under current task settings. Its architecture achieves an effective balance among model complexity, prediction accuracy, and resource efficiency—precisely our design objective. However, it's worth emphasizing that in more complex flow environments or higher-resolution datasets, hybrid attention modules like SE or CBAM may still offer advantages, with their inclusion determined on a case-by-case basis.

Conclusions

1. Model Contributions

This study proposes a low-complexity hybrid architecture—CNN-Transformer (CNN-T)—tailored for compressor blade aerodynamic performance prediction. Through comprehensive evaluation of different attention module integrations (SE, CBAM, multi-head attention) within convolutional frameworks, we demonstrate that embedding multi-head self-attention into CNN backbone networks significantly improves prediction accuracy while maintaining computational efficiency. The proposed CNN-T model achieves MAPE as low as 0.98%, surpassing more complex architectures like SS-CNN-T and SAM-CNN-T.

Our research reveals two key architectural insights: (1) Low-complexity convolutional kernels (3$×$3) combined with attention mechanisms can effectively capture both local and global information in aerodynamic characteristics. (2) Although module stacking may theoretically offer benefits, excessive module

stacking can lead to performance degradation due to the relatively structured and smooth nature of aerodynamic datasets.

These findings emphasize the importance of simplicity and task-specific design when developing efficient deep learning models for blade performance prediction.

2. Prediction Capability

The CNN-T model demonstrates strong generalization capability and accuracy when applied to blades with well-defined geometric profiles and steady flow conditions. Experimental validation proves that CNN-T excels at learning nonlinear relationships between key aerodynamic inputs (Mach number, Reynolds number, angle of attack) and outputs (such as velocity and pressure distributions).

Compared to traditional CFD-based methods, CNN-T improves prediction speed by one to two orders of magnitude, making it particularly suitable for real-time or large-scale design evaluation scenarios in subsonic environments. Consequently, this model provides a practical, efficient, and scalable solution for integrating deep learning into compressor blade aerodynamic design workflows.

References

1. Wang, H. F. (2024). Key technologies for collaborative design of high-performance fighter aircraft and engines. *Acta Aeronautica et Astronautica Sinica*, 45(05), 33-54.

2. Pinto, R. N., Afzal, A., D'Souza, L. V., Ansari, Z., & Mohammed Samee, A. (2017). Computational fluid dynamics in turbomachinery: a review of state of the art. *Archives of Computational Methods in Engineering*, 24(3), 467-479.

3. Wang, Y., Liu, T., Zhang, D., & Xie, Y. (2021). Dual-convolutional neural network based aerodynamic prediction and multi-objective optimization of a compact turbine rotor. *Aerospace Science and Technology*, 116, 106869.

4. Ma, Y. L., Du, Z., Xu, Q. Y., & Qi, J. H. (2024). Flow field reconstruction of compressor blade cascade based on deep learning methods. *Aerospace Science and Technology*, 155, Article 109637.

5. Wang, S. M., Chi, Z. D., Li, H. F., Wang, Q., Yan, W., & Jiang, B. (2024). Rapid identification and early warning of axial compressor stall based on multiscale CNN-SVM-FC model. *Aerospace Science and Technology*, 155, Article 109604.

6. Li, J., Deng, Y., Zhang, X., Wang, W., Fan, B., & Peng, F. (2024). Instability prediction under distorted inflow based on deep learning neural networks in an axial flow compressor. *Physics of Fluids*, 36(12).

7. Fan, Z. G., Wu, Y. T., Ba, D., Zhang, M., Liu, Y., & Du, J. (2025). Design optimization of axial slot casing treatment and blade in an axial compressor based on deep learning and reinforcement learning. *Aerospace Science and Technology*, 162, Article 110209.

8. Liu, Y. Y., Shen, J. X., Yang, P. P., & Yang, X. W. (2025). A CNN-PINN-DRL driven method for shape optimization of airfoils. *Engineering Applications of Computational Fluid Mechanics*, 19(1), Article 2445144.

9. Wang, L. Y., Xu, J., Luo, W., Luo, Z. H., Xie, J. H., Yuan, J. P., & Tan, A. C. C. (2022). A deep learning-based optimization framework of two-dimensional hydrofoils for tidal turbine rotor design. *Energy*, 253, Article 124130.

10. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

11. Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). Cbam: Convolutional block attention module. *Proceedings of the European conference on computer vision (ECCV)*.

12. Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*.

13. Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, 54(10s), 1-41.

14. Dosovitskiy, A. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

15. Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

16. Niu, Y., Zhao, K. N., Yang, Y. J., Yao, M. H., Wu, Q. L., Bai, B., & Ma, L. (2024). Integration of deep learning and computational fluid dynamics for rapid aerodynamic force prediction of compressor blades. *Physics of Fluids*, 36(10), Article 1.

17. Wu, M. Y., Wu, Y., Hua, Y., Chen, Z. H., Wu, W. T., & Aubry, N. (2025). Fast Aerodynamic Performance Prediction for Airfoils across Multiple Reynolds Numbers Using Deep Learning Method. *Journal of Aerospace Engineering*, 38(1), Article 040.

Note: Figure translations are in progress. See original paper for figures.