POLITECNICO DI TORINO

Master of Science in Automotive Engineering

Master Thesis

# Data-driven modeling of Dual-Dilution Combustion in Advanced Spark Ignition engines using Machine Learning


Politecnico di Torino

**Supervisors**
Prof. Federico MILLO
Dr. Andrea PIANO
Luigi TRESCA

**Candidate**
Alessandro SERAFINI

October 2025

# Abstract

Nowadays, it is fundamental to have fast and reliable virtual tools to accelerate the development of robust and efficient engines, in order to meet the increasingly stringent regulations introduced by the European Commission on both pollutant and greenhouse gas (GHG) emissions. In particular, when an ultra-lean dual-dilution approach is implemented as in this project, conventional 0D/1D CFD simulation software can be difficult and highly time-consuming to calibrate for achieving a reliable combustion model. Therefore, Machine Learning (ML) and Artificial Neural Network (ANN) can represent a valid alternatives to predict the combustion profile under these challenging conditions.

The aim of this thesis is to establish three different ML and ANN models for a ultra lean dual-dilution engine, based on experimental data obtained during the development of the PHOENICE (PHev towards zerO EmissioNs & ultimate ICE efficiency) H2020 project. More in detail, a fully connected neural network was designed to predict the Wiebe parameters, while a Gaussian Process Regression (GPR) model and an additional fully connected neural network were both developed to directly predict the burn rate curves.

These three models were able to capture the non-linear relationship between the core control variables and combustion with good accuracy, predicting the main combustion characteristics such as Mass Fraction Burned (MFB-10, MFB-50) and the combustion duration (MFB10–75). The best performing model, the fully connected neural network designed for burn rate prediction, was able to reach regression values between 0.9299 and 0.9953, while the root mean square errors between the ANN predicted and the experimental measurements were within the range of 0.57–0.90 °CA

# Contents

# List of Tables

# List of Figures

# Acronyms

**AI** Artificial Inteligence.

**ANN** Artificial Neural Network.

**BSFC** Brake Specific Fuel Consumption.

**BTE** Brake Thermal Efficiency.

**CI** Compression Ignition.

**CR** Compression Ratio.

**DDCA** Dual Dilution Combustion Approach.

**DI** Direct Injection.

**EC** European Commission.

**EGR** Exhaust Gases Recirculation.

**EU** European Union.

**GHG** Greenhouse gases.

**GP** Gaussian Process.

**GPR** Gaussian Process Regression.

**ICE** Internal Combustion Engine.

**ITE** Indicated Thermal Efficiency.

**IVC** Intake Valve Closure.

**IVO** Intake Valve Opening.

**ML** Machine Learning.

**MRE** Mean Relative Error.

**NCA** Neighborhood Component Analysis.

**NN** Neural Network.

**PFI** Port Fuel Injection.

**PFP** Peak Fired Pressure.

**RMSE** Root Mean Squared Error.

**SA** Spark Advance.

**SI** Spark Ignition.

**VNT** Variable Nozzle Turbine.

**VVA** Variable Valve Actuation.

# Introduction

Nowadays, increasingly stringent regulations imposed by the European Commission (EC), aimed at reducing emissions of air pollutants (CO, HC, $NO_x$, and PM) and Greenhouse gases (GHG) (mainly carbon dioxide, $CO_2$), are driving innovation in the automotive sector.

After the voluntary commitment by the automotive industry to achieve 140 g/km of $CO_2$ emissions by 2008 had failed [9], the European Union (EU) introduced mandatory $CO_2$ standards for passenger cars in 2009.

The legislation currently in force is an evolution of the European Commission's *Fit for 55* package, which is a series of regulations aimed at reducing EU greenhouse gas emissions by at least 55% by 2030, compared to 1990 levels [1]. Over the years, this plan has undergone numerous modifications, with the timeline for achieving carbon neutrality being progressively anticipated. The most recent amendment was formally adopted on March 28, 2023, introducing a 100% $CO_2$ reduction target for new passenger cars by 2035 and strengthening the 2030 intermediate target from –37.5% to –55%, relative to 2021 baseline levels [1]. The evolution of the legislation is summarized in *Figure 1*.

Considering the stringency of the targets and the limited time to achieve them, it is fundamental to accelerate the development phase of new technologies. In this context, the data-driven approach can play a fundamental role.

Several studies have been carried out in recent years on the use of Artificial Neural Network (ANN) and Machine Learning (ML) models to predict the performance and emissions of Internal Combustion Engine (ICE) [10, 11]. Among these, the second study [11] focused mainly on the prediction of Brake Specific Fuel Consumption (BSFC), Brake Thermal Efficiency (BTE), CO (%) and HC (ppm), using data from

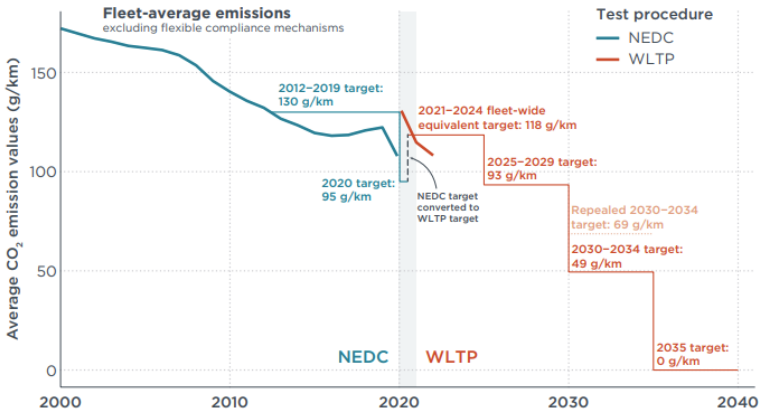



Figure 1: EU $CO_2$ legislation evolution - [1]

96 steady-state test runs with a three-layer ANN, and achieved a Mean Relative Error (MRE) in the range of 1.41–6.66%.

Additionally, a recent study on combustion modelling for modern spark-ignition engines based on data-driven methods [12] was carried out. This research aimed to evaluate the performance of two ANNs: one designed to predict sampled MFB points and the other to predict fitted Wiebe coefficients, using a total of 1258 cases. The first model outperformed the second, achieving $R^2 > 0.95$ in more than 95% of the cases in both the training and validation datasets, with the optimal configuration of three hidden layers containing 16–9–16 neurons.

The present dissertation aims to develop machine learning models for combustion modelling in Spark Ignition (SI) engines, with the goal of supporting the development of next-generation high-efficiency ICEs. Three models will be presented: two ANN models, one for predicting Wiebe coefficients and another for burn rate profile prediction, and one Gaussian Process Regression (GPR) model, also for burn rate prediction.

# Chapter 1

# Theoretical Background

The aim of this chapter is to provide an overview of the theoretical foundations of the project. With a particular focus on combustion modeling and the fundamentals of machine learning.

## 1.1 Combustion Modelling

The combustion process in Internal Combustion Engines (ICEs) is the key process that converts chemical energy of the fuel into mechanical work, directly influencing engine performance, efficiency, and emissions [13]. Combustion is not a single process, but varies significantly depending on the engine type. Among the two main categories - Spark Ignition (SI) and Compression Ignition (CI) engines — this work focuses exclusively on the former.

In SI engines, air and fuel are premixed, with a defined air-fuel (A/F) ratio, before the start of combustion. For Port Fuel Injection (PFI) the mixing phase occurs in the intake system; on the other hand, for Direct Injection (DI) systems it occurs directly inside the combustion chamber since the fuel is injected directly into the cylinder during the intake stroke. The first strategy promotes better mixture homogeneity, the second offers the possibility to implement charge stratification strategies, in addition to the homogeneous charge, which can improve combustion efficiency and emission control [14].

Combustion is triggered near the end of compression by an electric discharge

from a spark plug. This discharge ignites first kernel, which radially propagates outward. The flame front advances due to intense heat transfer from the hot, burned gases to the adjacent layers of fresh, unburned mixture. Under normal operating conditions, the combustion process in SI engines can be divided into three distinct phases:

- **Development phase**: First kernel is ignited by the spark, and grows until the flame front is developed. Approximately 10% of the total mass is burned.

- **Rapid burning phase**: Turbulent flame front propagates through the combustion chamber until it reaches the walls. This phase is responsible for the bulk of the energy release. From 10% to 90% of the charge is burned.

- **Termination phase**: Last portion of the charge, about 10%, completes its oxidation process. The chemical energy is converted into heat.

Considering the crucial role that combustion modelling plays in obtaining reliable simulation results for engine performance evaluation, it is fundamental to develop accurate and robust combustion models to support the design of next-generation high-efficiency ICEs.

For what concern 0D/1D CFD simulations, there are two main models that are widely used: the Single-Zone model and the Multi-Zone model. Each of these will be described below.

### 1.1.1 Wiebe Function

One example of a Single-Zone model is the Wiebe function, which is an empirical model that relies on experimental data and does not directly account for all the reactions involved during the process. It estimates the mass fraction burned as a function of engine position, following an S-shaped curve (reported in *Figure 1.1*), which rises from zero to one, with the interval defining the combustion duration [13].

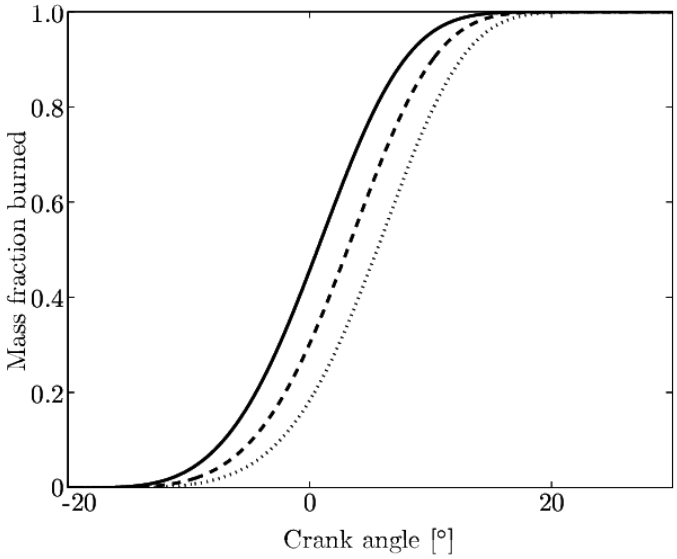



Figure 1.1: Wiebe function shape - [2]

The curve is evaluated by using the following equation (*1.1*):

$$x_b(\theta) = 1 - \exp\left(-a\left(\frac{\theta - \theta_0}{\Delta\theta}\right)^{m+1}\right) \tag{1.1}$$

Where:

- $a$: efficiency parameter of the Wiebe function
- $m$: form factor of the Wiebe function
- $\theta$: crank angle
- $\theta_0$: start of combustion
- $\Delta\theta$: combustion duration

The Wiebe function parameters are tailored to specific engines and operating conditions. Although it offers a simple and robust representation, it has inherent limitations in accurately capturing the full complexity of combustion dynamics.

### 1.1.2 Two-Zone combustion Model

A multi-zone combustion model, instead, provides a more accurate physical alternative compared to empirical model such as the Wiebe function.

This approach divides the combustion chamber into several thermodynamic zones to more accurately simulate the in-cylinder combustion dynamics. In contrast to single-zone models - e.g., the Wiebe function - which assume uniform properties throughout the chamber, multi-zone approaches treat different regions as distinct entities that evolve independently over time.

Two-Zone combustion model, which is one of the most common multi-zone model used in engine simulation, divides the combustion into two zones: unburned and burned zone (*Figure 1.2*).

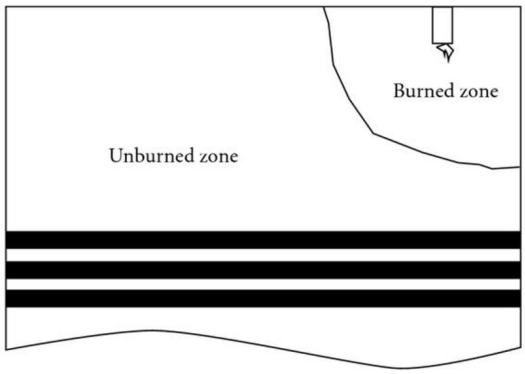



Figure 1.2: Burned and Unburned zone - [3]

At the beginning of combustion, all the species in the cylinder are in the unburned zone, including residuals and EGR. Once the spark ignites the mixture, a small reaction zone is formed around the spark plug. As combustion progresses, the model transfers part of the unburned mixture into the burned zone based on a defined burning rate. The burned zone is then assumed to be in chemical equilibrium, where the composition depends on its temperature and pressure. The model evaluates the internal energy of each species in the burned zone, sums them, and applies energy conservation balance between zones to update temperatures and cylinder pressure. This method captures the thermodynamic changes in detail, providing more accurate predictions of pressure and emissions than simpler models

[15].

## 1.2 Artificial Intelligence Fundamentals

### 1.2.1 Introduction

Artificial Inteligence (AI) is a broad branch of computer science that enables computers and machines to emulate human-like intelligence by understanding, learning, making decisions, exhibiting creativity, and operating autonomously [16].

Although the term *Artificial Intelligence* was coined in 1956 by scientist John McCarthy, the theory of machines with human-like intelligence dates back much further. One of the earliest ideas about "machine learning" emerged in 1914, when the Spanish inventor Leonardo Torres y Quevedo built an electromechanical machine capable of playing chess endgames without human intervention.

Despite initial enthusiasm, progress slowed during the 1970s and 1980s due to limited computing power and overestimated capabilities. This period, known as the "AI winter", was characterized by widespread disillusionment as the technology failed to meet investor and public expectations.

In the 1990s, AI research had a renaissance and in the 2010s it intensified, driven largely by increased computational power, the availability of large datasets, and algorithmic innovations. In 2012, the deep learning model AlexNet achieved a breakthrough in image classification, marking a turning point for the modern era of AI [17].

### 1.2.2 Machine Learning

Machine learning (ML) is a branch of Artificial Intelligence (AI) that focuses on the development of algorithms capable of learning patterns from data and making predictions or decisions without being explicitly programmed for specific tasks.

Nowadays, machine learning is widely used in everyday life: it powers virtual personal assistant or voice assistance, like Apple's Siri or Amazon's Alexa, which can communicate with humans by recognizing speech and carrying out the requested actions; it is also implemented in Google's Gmail to automatically categorize email into different folders like Primary, Social, and Promotional, as well as to identify

and filter spam.
An additional application of machine learning is in healthcare, where models are trained to classify tumors, find bone fractures that are hard to see, and detect neurological disorders [18].

Machine learning models are employed in several disciplines and, since algorithms are very different from each others they are generally classified into three main categories based on the nature training data and learning approach [19][20]:

- **Supervised Learning**: the data are labeled and divided into input and output sets. These labeled data are used to train algorithms, enabling the model to evaluate their performance through accuracy metrics and learn over the time by progressively improving their predictions.
- **Unsupervised Learning**: the data are unlabeled and generally only input features are provided to the model. These algorithms aim to uncover hidden patterns or relationship within the input data, grouping similar data points.
- **Reinforcement Learning**: the model interacts with a dynamic environment and learns by trial and error to optimize a given objective. It receives feedback in the form of rewards based on the outcomes of its actions. By maximizing the cumulative reward, the model learns the best strategies to reach its goal.

Models can also be classified further according to the structure of their output, as shown in *Figure 1.3*.

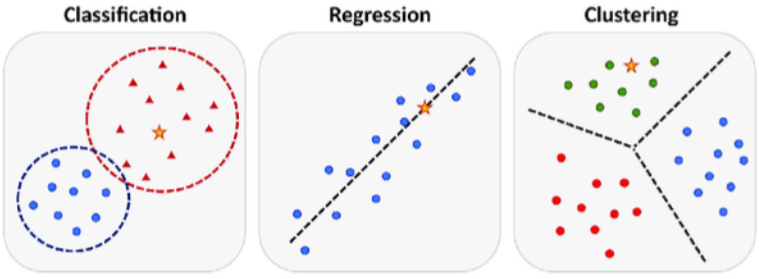



Figure 1.3: Machine learning categories - Output Structure

- **Regression**: It is a supervised learning model trained to understand the relationship between input features and outputs, where the outputs are continuous numerical variables.

- **Classification**: It is another type of supervised learning model, trained to accurately assign test data to specific categories; the outputs can only take discrete values.
- **Clustering**: It is an unsupervised learning technique, trained to group unlabeled data based on similarities or differences, where the output consists of clusters rather than predefined labels.

### 1.2.3 Model complexity

The complexity of model is a crucial factor that determines its ability to capture patterns in the data. It is crucial find the optimal level of complexity, otherwise if the model is too simple, it could fail in capturing important patterns. On the other hand, if it is too complex, it learns to fit also the noise in the data rather than true patterns. Two key concepts commonly used to describe model complexity are:

- **Bias**, which denotes the simplifying assumptions introduced during training to make the learning process easier, often at the cost of ignoring part of the underlying complexity.
- **Variance**, which represents the degree by which the model is affected by variations in the training data.

As can be seen from *Figure 1.4*, at both extremes of complexity, too simple or too complex, the error tends to increase dramatically. It is therefore essential to design a model that is sufficiently complex to capture the true patterns, but not so complex to fit the noise.

In other words, if a model has high bias it is said to be underfitting, whereas if it has high variance it is said to be overfitting. Both problems are related to the ability of a model to generalize to unseen data.

- **Overfitting** occurs when a model achieves high accuracy on the training data but fails to perform well on unseen data. It often arises when the model is too complex, as previously discussed, or as a result of high-dimensional datasets.
- **Underfitting** represents the opposite condition. In this case, the model is too simple to capture the essential patterns, leading to poor performance on both

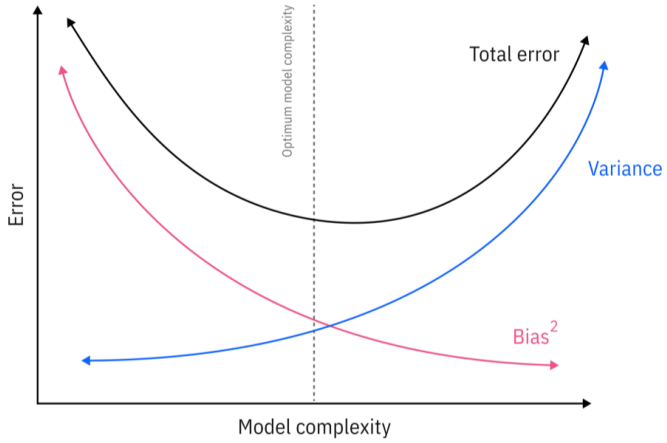



Figure 1.4: Model complexity trade off - [4]

training and testing sets. Typical causes include simplistic models, inadequate feature engineering, or insufficient training data.

Examples of underfitting, overfitting, and optimal fitting for both classification and regression problems are shown in *Figure 1.5*.

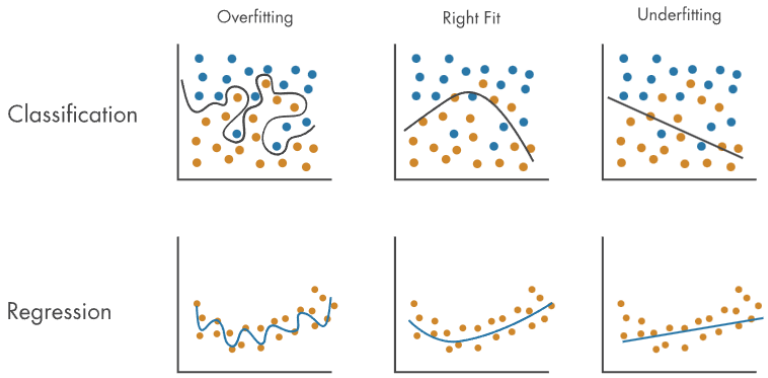



Figure 1.5: Underfitting vs Right fitting vs Overfitting - [5]

### 1.2.4 Feature selection

A feature is a measurable quality of the elements in a dataset, it is also known as attribute, since it describes the data [21]. Features can be:

- **Independent** variables, which are the inputs of the models.
- **Dependent** variables, which depend on independent ones.
- **Derived attributes**, which are compiled from multiple other features.

Additionally, they can be further categorized into:

- **Numerical** variables that are measurable, such as age and sizes.
- **Categorical** variables that are everything non-numerical, such as name and surname.

Feature selection is a branch of feature engineering [22], which is the process of transforming raw data into a machine-readable format, that aims to reduce the feature space by finding the most relevant to predict the targets. The benefits of feature selection consist of the following: improving model performance by removing irrelevant features that might negatively affect the accuracy level, reducing the risk of overfitting, and reducing computational costs and training time. Two main subgroups can be identified, depending on the technique used to select the most important features:

- **Supervised methods**: they use the target values to determine the most important features. These techniques can be further grouped into three main categories:
  - Filter methods, which evaluate the input feature only based on their statistical relationship with the target variable. They are fast and efficient; however, it does not consider feature interactions or model performance. Common example are *Pearson's correlation coefficient* and *Neighborhood Component Analysis*.
  - Wrapper methods, which select features by evaluating the performance of the machine learning model trained with different subsets of features. They are highly computational and time consuming; but, take into account feature interactions and model performance. Common examples include *Recursive Feature Elimination* and *Exhaustive Feature Selection*.

  - Embedded methods, which perform feature selection as part of the training process. They use regularization techniques, such as Lasso or Ridge, to penalize less relevant features.

- **Unsupervised methods**: they do not rely on target values, but instead analyze patterns and variance within the input data to identify relevant features. One example is *Principal Component Analysis* that reduces the dimensionality of large datasets by transforming correlated variables into a smaller set of uncorrelated variables called principal components.

### 1.2.5 Gaussian Process Regression

Gaussian Process Regression (GPR) is a non-parametric regression technique. Its flexibility and power are particularly exploited with problems involving continuous data, where the correlation between inputs and outputs is unspecified or highly complex. It is based on the Gaussian Process (GP) model, which is widely used in machine learning and statistics thanks to its main characteristics, which are:

- **Non-parametric nature**: it is capable of adapting to data complexity, as it does not rely on a fixed set of parameters.
- **Probabilistic predictions**: it is capable of quantifying the accuracy of predictions, as it delivers a probabilistic distribution.
- **Interpolation and smoothing**: it is capable of handling noisy sampled data, as it provides effective smoothing and interpolation.
- **Marginalization of hyperparameters**: it is capable of simplifying the model, as it marginalizes over hyperparameters and removes the need for explicit tuning.

Gaussian Processes (GPs) are based on a few main elements that work together to set initial assumptions, capture patterns in the data, and update those assumptions once new observations are introduced. The first element is the *mean function*, which represents the expected value of the function at each input, often set to zero by default. The most important element is the *covariance*, or *kernel*, *function*, which measures similarity between inputs and determines how the model captures

patterns. Finally, two types of distributions are involved: the *prior distribution*, which combines the mean and covariance functions to encode assumptions before any data are observed, and the *posterior distribution*, which updates the prior distribution once data are available, providing predictions along with uncertainty estimates [23].

Some of the most common basic kernel functions are presented below [24]:

- **Radial Basis Function (RBF) kernel**: it is characterized by a key hyper-parameter ($l$), called the length-scale parameter. This parameter controls how quickly the correlation between two points decreases as their distance, $d(x_i, x_j)$, increases. Therefore, if $l$ is large, distant points remain highly correlated, and as a result the function will be smoother. Conversely, if $l$ is small, the correlation between two points decreases more rapidly, and the function will exhibit larger fluctuations.

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right) \tag{1.2}$$

- **Matérn kernel**: it is a generalization of the RBF kernel. Thus, it has not only $l$ as a parameter, but also an additional term $\nu$, which controls the smoothness of the resulting function. If $\nu$ has a small value, the resulting function will be less smooth. In contrast, as $\nu \to \infty$, the kernel becomes equivalent to the RBF.

$$k(x_i, x_j) = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\frac{\sqrt{2\nu}}{l}d(x_i, x_j)\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}}{l}d(x_i, x_j)\right) \tag{1.3}$$

- **Rational Quadratic kernel**: it is an extension of the RBF kernel that introduces a parameter $\alpha$, called the scale-mixture parameter. This parameter controls how the kernel combines different smoothness levels, making it useful when the function may not have the same smoothness everywhere in the input space.

$$k(x_i, x_j) = \left(1 + \frac{d(x_i, x_j)^2}{2\alpha l^2}\right)^{-\alpha} \tag{1.4}$$

Different basic kernels can be combined with each other, using addition and

multiplication, to create new kernels.

### 1.2.6 Artificial Neural Networks

The last machine learning model, that is presented in this work is called Artificial Neural Networks (ANNs), mainly simply called Neural Networks (NNs). This type of model tries to reproduce the operating principle of biological neurons in the human brain to mimic the human thinking approach. This technique was developed to overcome problems that could not be solved with traditional computer programming or other machine learning models, such as speech or image recognition, although they are easily solvable by the human brain [25].

Traditional ANNs, usually composed of one input layer, a single hidden layer, and one output layer, often have limited capacity to manage complex tasks due to small dimensions. For this reason, a new branch of AI and ML has emerged, called deep learning. Deep learning is essentially an artificial neural network model that uses bigger architecture and deeper networks [26].

As mentioned before, NNs are inspired by the functionality of human brain, which works thanks to elementary units, called neurons, connected to each other, *Figure 1.6*, and exchanging information via electrical impulse. In the same way, neural networks are composed by artificial neurons, also called perceptrons, that are connected within a network, *Figure 1.7*, and exchange information using numerical values. Despite the same working principle, it is important to highlight that ANNs are only simplified mathematical models derived from biological processes, rather than accurate replications of them [27].

Focusing on artificial neurons, *Figure 1.8*, which aim to replicate the behavior of biological ones, the artificial one is composed of an input signal, a processing unit that performs a mathematical operation, and an output that is then transmitted to the following neurons.

Inside the processing unit, the following sum is performed, which returns the *pre-activation function*, $a$:

$$a = \bar{x} \cdot \bar{W} + b = \sum_{k=1}^{n} W_k x_k + b \tag{1.5}$$

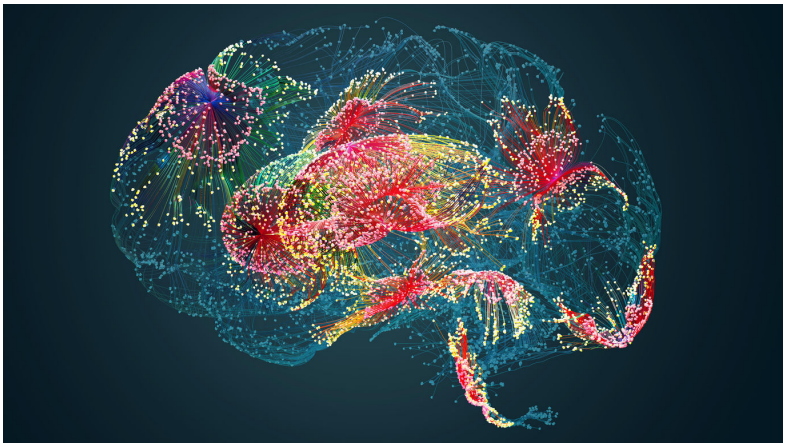


Figure 1.6: Picture of biological neurons connection - [6]

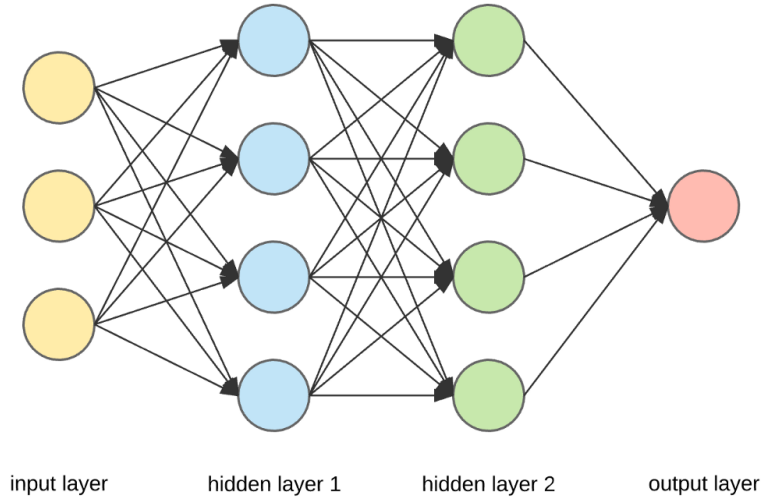



Figure 1.7: Example of artificial neurons connection - [7]

Where:

- $\bar{x} \in \mathbb{R}^N$ is the *input vector*, that represent the outputs generated by the other $N$ connected neurons.
- $\bar{W} \in \mathbb{R}^N$ is the *weight vector*, which defines the influence that each input has on the corresponding output.
- $b$ is the *bias*, an additional parameter, independent of the inputs, that serves as a constant offset of the weighted sum before it is passed to the activation

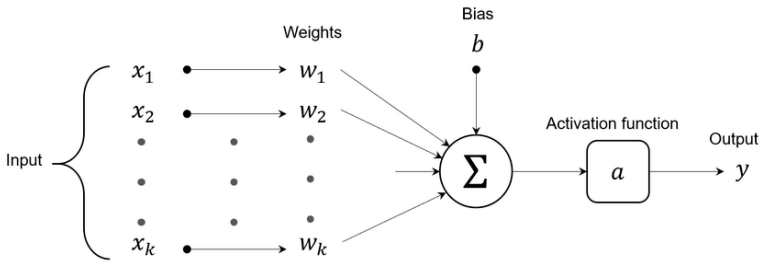



Figure 1.8: Artificial Neuron

function.

After that, a function, called *activation function* $f(\cdot)$ is applied to the weighted sum. This introduces non-linearity and makes possible to model complex relationships between inputs and output.

$$y = f(a) = f\left(\sum_{k=1}^{n} w_k x_k + b\right) \tag{1.6}$$

Different activation functions have been introduced in the literature, and each of them gives the neuron specific features, making it more suitable for certain applications [28].

Some of the most common ones are presented and illustrated below:

- Hard limiter or threshold function:

$$f(a) = \begin{cases} k, & a \geq 0 \\ 0, & a < 0 \end{cases} \tag{1.7}$$

- Rectified linear unit function:

$$ReLU(a) = \begin{cases} a, & a \geq 0 \\ 0, & a < 0 \end{cases} \tag{1.8}$$

- Sigmoid function:

$$\sigma(a) = \frac{1}{1 + e^{-a}} \tag{1.9}$$

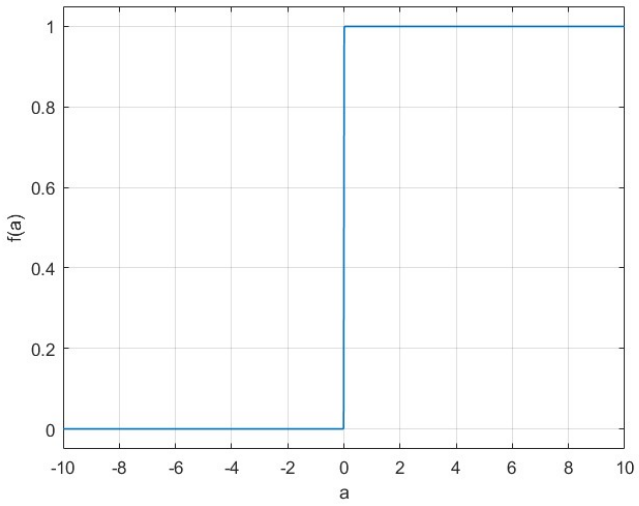



Figure 1.9: Threshold function

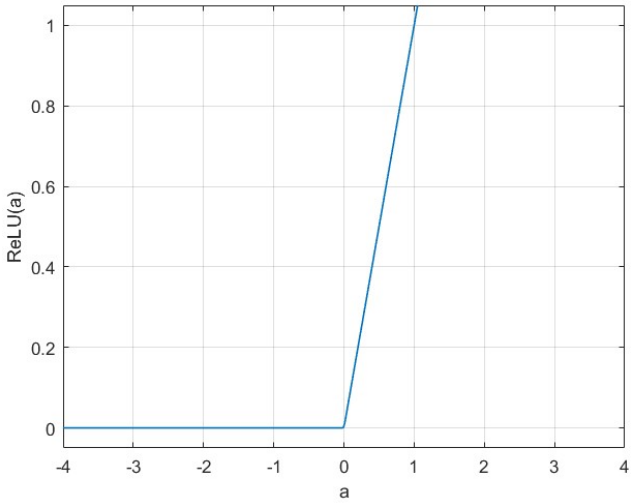



Figure 1.10: ReLU function

- Hyperbolic tangent function:

$$\tanh(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}} \tag{1.10}$$

For a neural network to work properly, it is essential that the weights and

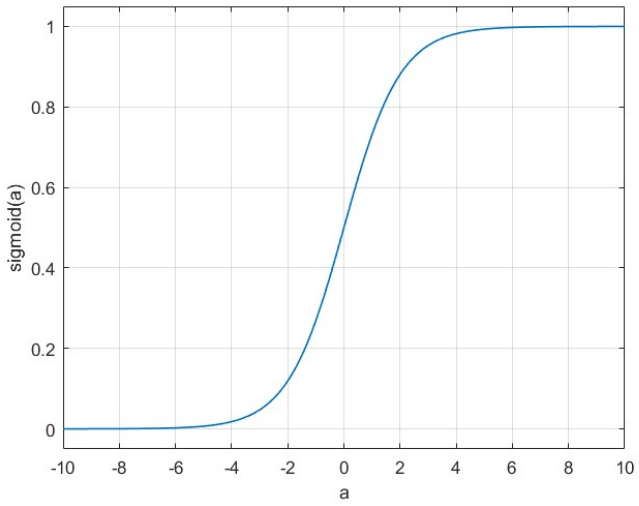



Figure 1.11: Sigmoid function

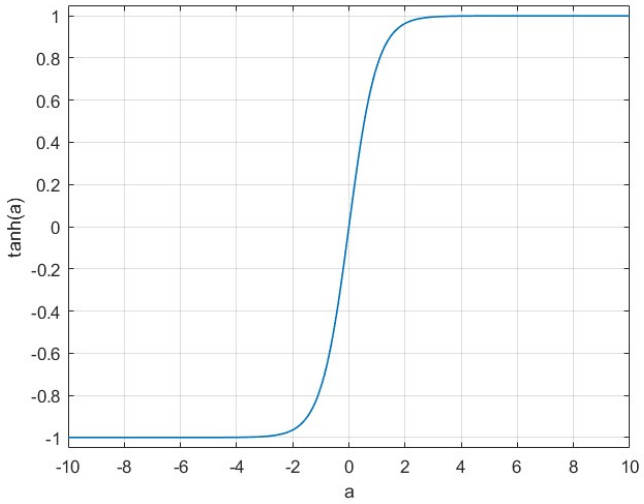



Figure 1.12: Hyperbolic tangent function

biases are chosen appropriately. Fortunately, optimal values are determined during the training process, where the model autonomously learns complex relationships between inputs and outputs without the need for manual adjustment by the user.

### Neural Network architecture

As mentioned earlier, ANNs are composed of multiple layers, each containing a set of neurons that connect only to neurons in other layers. The number of layers and neurons defines the architecture of the network.

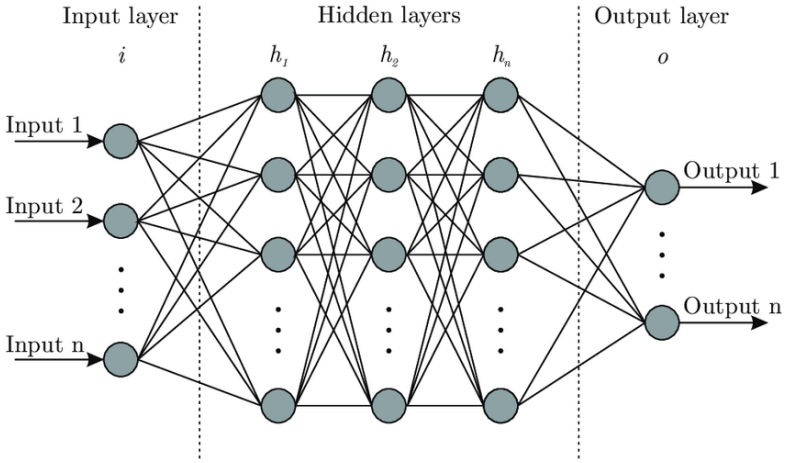



Figure 1.13: Neural Network architecture

The layers of the network can be divided into three main categories *Figure 1.13* [29]:

- **Input layer**: it is the first layer, and the number of neurons corresponds to the number of features of the input data. This layer does not perform any computational processes, but simply passes the data to the hidden layer.

- **Hidden layers**: these are all the layers between the input and output layers. The number and size of the hidden layers can vary depending on the complexity of the problem. This is where most of the computational cost of the network is concentrated, since each hidden layer applies a set of weights and biases to the input data.

- **Output layer**: this is the last layer of an ANN, responsible for producing the output predictions. The number of neurons corresponds to the number of classes in a classification task, or to the number of outputs in a regression problem.

### Learning algorithms for neural networks

Learning algorithms represent the fundamental mechanism that allows artificial neural networks (ANNs) to set their internal parameters and improve performance through experience. The most common approach is based on gradient descent, where the algorithm computes the gradient of a cost function with respect to the network weights and updates them in order to minimize the prediction error. Variants such as stochastic gradient descent (SGD) and mini-batch gradient descent enhance computational efficiency by performing weight updates on subsets of the training data. Furthermore, advanced optimization strategies, including Momentum, RMSprop, and Adam, have been introduced to accelerate convergence and improve stability by dynamically adapting the learning rate and overcoming local minima. The process of backpropagation provides an efficient way to propagate error signals through the network, ensuring that weight adjustments are distributed across all layers. Overall, these algorithms are essential for enabling ANNs to capture complex nonlinear relationships and generalize effectively to unseen data [30].

# Chapter 2

# Case study

## 2.1 Engine specification

The study was carried out using performance data obtained during the development of the European Horizon 2020 PHOENICE project, based on a state-of-the-art 4-cylinder, 1.3L turbocharged direct injection spark-ignition engine [31]. The engine is characterized by a high stroke-to-bore ratio, a compact 4-valve combustion chamber with side-mounted 200 bar fuel injection system, a MultiAir Variable Valve Actuation (VVA) [32] system and a cylinder head with integrated exhaust manifold.

With the aim of achieving a peak Indicated Thermal Efficiency (ITE) of 47% the Dual Dilution Combustion Approach (DDCA) [33] was implemented, combining homogeneous lean combustion and cooled low-pressure Exhaust Gases Recirculation (EGR), with high Compression Ratio (CR). To support this strategy, several breakthrough technologies were integrated to the baseline engine, particularly focusing on the combustion system.

The main upgrades, as shown in *Figure 2.1*, include the redesign of the piston, which increased the CR from the baseline value of 10.5 to 13.6. Together with the redesigned intake port geometry, this allowed the exploitation of the Swumble concept, improving turbulence levels and supporting flame propagation under the challenging conditions of DDCA.

The boosting and fuel injection systems were also upgraded, introducing a 48V

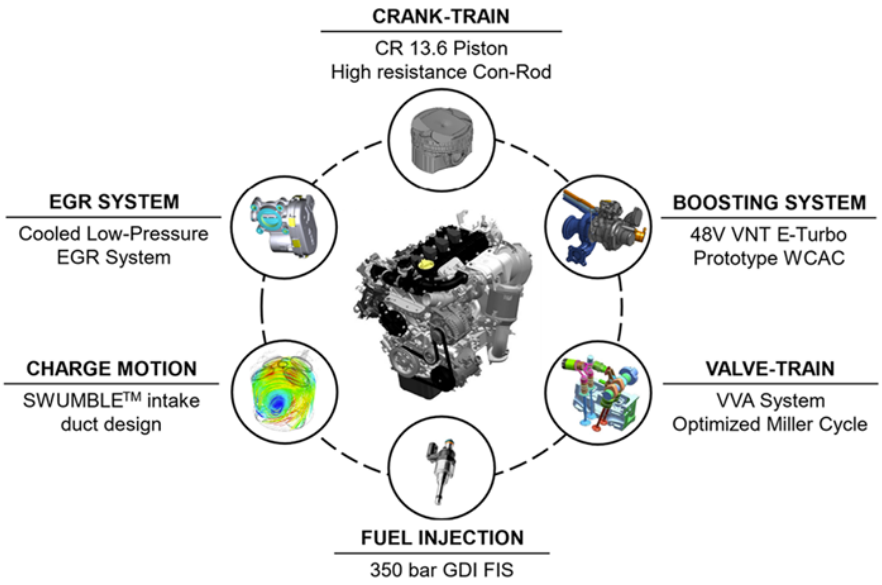



Figure 2.1: Breakthrough technologies adopted on PHOENICE engine

electrified turbocharger with a Variable Nozzle Turbine (VNT), which not only allows to mitigate turbo lag, but also enables energy recovery when the turbine produces more energy than required by the compressor. In addition, a fuel injection system capable of operating up to 350 bar was implemented.

Finally, the VVA system was fully exploited to implement aggressive Miller cycle strategies, reducing pumping losses. The integration of these technologies resulted in the specifications reported in *Table 2.1*

| **Engine Specifications** | |
|---|---|
| *Number of cylinders* | 4 |
| *Displacement* | 1332 $cm^3$ |
| *Bore x Stroke* | 70 mm x 86.5 mm |
| *Compression Ratio* | 13.6:1 |
| *Number of valves* | 16 |
| *VVA system* | MultiAir III (intake only) |
| *Turbocharging* | 48 V VNT E-Turbo |
| *Fuel Injection* | GDI (up to 350 bar) |
| *Ignition System* | Base production |
| *EGR System* | Cooled Low Pressure (LP) |
| *Rated Power (target)* | 100 kW @ 4500 RPM |
| *Rated Torque (target)* | 218 Nm @ 3500 RPM |

Table 2.1: PHOENICE Engine Specifications

## 2.2 Test matrix

As stated earlier, the data used in the elaboration of this dissertation were obtained during the development of the PHOENICE engine. In particular, they come from steady-state tests performed by IFPEN during the fine-tuning phase. These calibration tests were carried out at eleven engine operating points, selected to represent a broad range of conditions in terms of engine speed and load demand, as illustrated in *Figure 2.2*. The complete set of operating points is further summarized in *Table 2.2*.

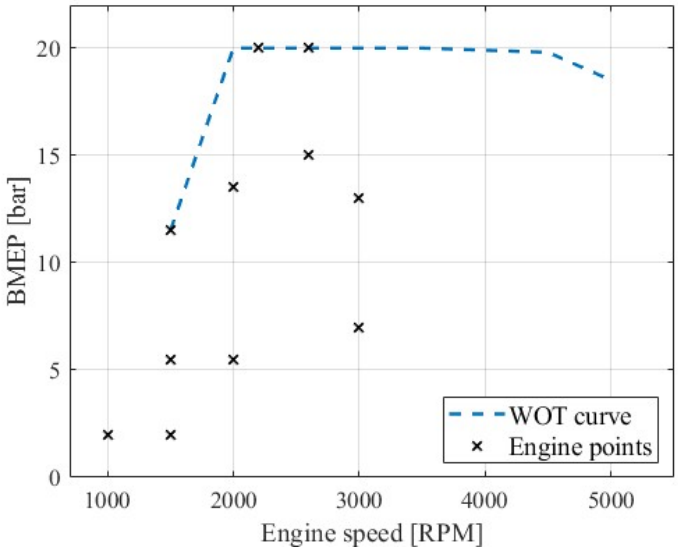



Figure 2.2: Engine operating points and full load curve

During the tuning process to characterize the influence of the dual dilution combustion approach, the reported operating points were tested at different values of air–fuel ratio ($\lambda$) and Exhaust Gas Recirculation (EGR) rates. In total, 101 operating conditions were obtained, each corresponding to a specific working condition defined by speed, load, EGR and $\lambda$. As examples, *Table 2.3* and *Table 2.4* present two representative combinations of operating points, corresponding to 1500 RPM $\times$ 5.5 bar BMEP and 3000 RPM $\times$ 7 bar BMEP, that will be used to support the discussion in the following chapter.

For each operating point, several data were acquired during the testing phase

| **Tested engine points** | |
|---|---|
| *Engine speed [RPM]* | *BMEP [bar]* |
| 1000 | 2 |
| 1500 | 2 |
| 1500 | 5.5 |
| 1500 | 11.5 |
| 2000 | 5.5 |
| 2000 | 13.5 |
| 2200 | 20 |
| 2600 | 15 |
| 2600 | 20 |
| 3000 | 7 |
| 3000 | 13 |

Table 2.2: Tested Engine Points

| **1500 RPM × 5.5 bar BMEP** | |
|---|---|
| *Air-fuel ratio [-]* | *EGR Rate [%]* |
| 1.00 | 0.90 |
| | 5.20 |
| | 10.2 |
| | 15.3 |
| | 18.9 |
| 1.11 | 0.70 |
| | 10.3 |
| | 15.0 |
| 1.25 | 0.60 |
| | 5.10 |
| 1.43 | 0.00 |

Table 2.3: 1500 RPM × 5.5 bar BMEP $\lambda$ and EGR sweeps

by IFPEN, including air flow rate, fuel flow rate, pressure and temperature in different locations of the intake and exhaust system, such as at compressor and turbine inlets and outlets. Additional recorded variables include brake specific fuel consumption, brake specific CO, HC, and NOx emissions, turbocharger speed, and Spark Advance (SA). Finally, it should be highlighted that the burn rate curves used in this project were obtained from a GT-Suite model previously calibrated in an earlier thesis, using experimental data provided by IFPEN.

| **3000 RPM × 7 bar BMEP** | |
|---|---|
| *Air-fuel ratio [-]* | *EGR Rate [%]* |
| 1.00 | 0.00 |
| | 5.20 |
| | 10.0 |
| | 15.2 |
| | 20.6 |
| | 21.5 |
| 1.11 | 0.00 |
| | 5.30 |
| | 9.80 |
| | 15.1 |
| | 20.1 |
| 1.25 | 0.00 |
| | 5.10 |
| | 10.0 |
| | 14.4 |
| 1.43 | 0.00 |
| | 5.20 |
| | 7.20 |

Table 2.4: 3000 RPM × 7 bar BMEP $\lambda$ and EGR sweeps

# Chapter 3

# Methodology

This chapter outlines the methodology followed throughout this project, starting with the first fundamental step common to all the developed models, the process of feature selection and the analysis of feature density distributions. It then provides a detailed analysis of each proposed model, starting with the baseline neural network trained to predict the parameters of the Wiebe function, followed by the application of Gaussian Process Regression (GPR) for the prediction of burn rate curves, and concluding with the description of the ANN model developed to directly predict the combustion profile.

## 3.1 Feature selection

As stated in *Section 1.2.4*, feature selection is essential to ensure good performance of machine learning models, while also reducing the computational cost required by computers. Indeed, when working with large datasets and numerous features, many may be irrelevant to the target objective. Including such features in the dataset can degrade model performance, with possible issues related to overfitting or underfitting, as anticipated in *Section 1.2.3*.

In this project, the selection of the most relevant inputs for the models was carried out through a hybrid approach that combined mathematical models with engineering knowledge. Some of the data acquired by IFPEN also included measurements that were either irrelevant for the analysis or directly linked to the target prediction,

such as emissions or Peak Fired Pressure (PFP).

The initial set of 12 selected features is reported in *Table 3.1*:

| **Initial Features** | |
|---|---|
| Engine speed | [RPM] |
| Turbo speed | [RPM] |
| Exhaust gas recirculation | [%] |
| Air-fuel ratio -$\lambda$ | [-] |
| Intake pressure | [bar] |
| Intake temperature | [K] |
| Intake valve opening | [deg CA] |
| Intake valve closure | [deg CA] |
| Injected fuel mass | [mg/cycle] |
| Injection duration | [deg CA] |
| Start of injection | [deg CA] |
| Spark advance | [deg CA] |

Table 3.1: Initial set of features after first human screening

Starting from this set, two supervised feature selection methods were implemented: *Pearson's correlation coefficient* and *Neighborhood Component Analysis (NCA)*. Both of these are filter-types techniques, meaning that they evaluate the importance of the inputs only based on their statistical relationship with the target variable. Since it was not possible to use the full combustion curve, three main combustion parameters were considered:

- *MFB-10*, the crank angle at which 10% of the total fuel mass is burned;
- *MFB-50*, the crank angle at which 50% of the total fuel mass is burned;
- *MFB-1075*, the combustion duration, defined as the crank angle interval between 10% and 75% of the total fuel mass burned.

It should be noted that MFB-1075 was selected instead of MFB-1090, as under extreme conditions, especially in dual dilution operation, the achievement of 90% mass fraction burned was either not possible or substantially delayed.

Since the models are sensitive to the scale of individual variables, both the inputs and the targets were normalized to enable a fairer comparison of their relative influence.

### 3.1.1 Neighborhood Component Analysis

This method evaluates the importance of the features by assigning a score with respect to the target value. In the case of regression, *Neighborhood Component Analysis* (NCA) learns feature weights by minimizing the prediction error of a nearest-neighbor regressor. Features with higher scores are considered more relevant for predicting the target, while those with negligible weights can be excluded from the model.

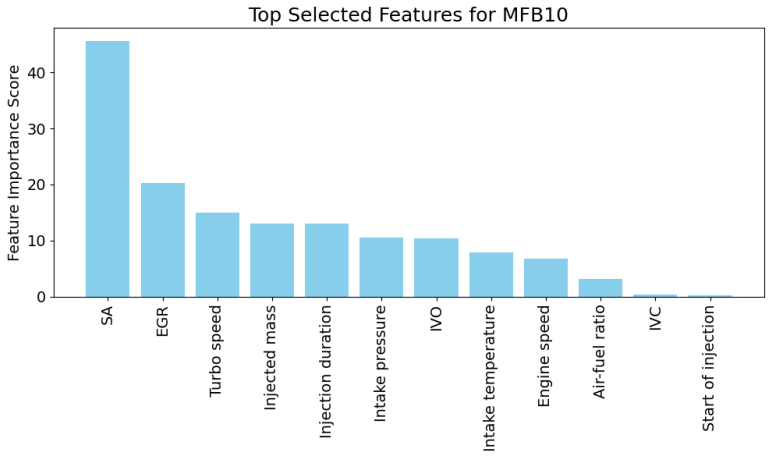



Figure 3.1: Feature scores - target MFB10

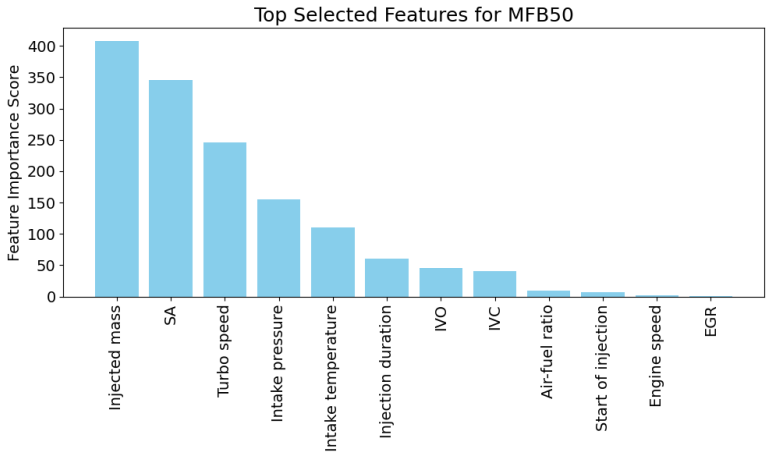



Figure 3.2: Feature scores - target MFB50

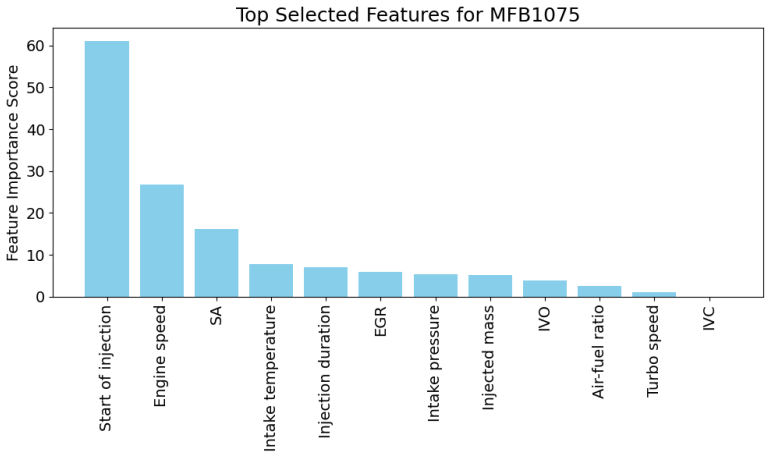



Figure 3.3: Feature scores - target MFB1075

*Figures 3.1–3.3* illustrate the feature scores obtained for the three combustion targets: MFB10, MFB50, and MFB1075. The following observations can be made:

- **Intake Valve Opening (IVO)** and **Intake Valve Closure (IVC)** show low relevance in all three cases, as they are optimized and fixed at each operating point to maximize the Millerization effect.
- **Start of injection** has negligible impact on two of the three targets (MFB10 and MFB50), but plays a major role in determining the combustion duration (MFB1075).
- **Injected fuel mass** and **injection duration** display comparable scores, except for MFB50, where the injected fuel mass becomes the most relevant feature.
- **Air–fuel ratio** exhibits low relevance across nearly all targets.
- **EGR** shows some influence on MFB10, but has limited importance for MFB50 and MFB1075.
- **Intake temperature** and **intake pressure** consistently hold intermediate importance.
- **Spark advance** (SA) ranks among the top three most influential features across all cases.

- **Engine speed** is almost irrelevant for MFB50, but, as expected, it significantly affects both MFB10 and MFB1075.
- **Turbo speed** shows a small contribution, limited mainly to the combustion duration (MFB1075).

### 3.1.2 Pearson's correlation coefficient

Because Neighborhood Component Analysis (NCA) does not account for interactions between different features, an additional technique was applied. Pearson's correlation coefficient, *Equation 3.1*, was selected to validate the NCA results and to verify whether some features are correlated with each other and therefore potentially redundant during the training process.

Pearson's correlation coefficient ($r$) is a statistical measure that quantifies the strength and direction of the linear relationship between two continuous variables. Its value ranges from $-1$ to $+1$ [34], where:

- $+1$ indicates a perfect positive linear relationship, as one variable increases, the other increases proportionally.
- $-1$ indicates a perfect negative linear relationship, as one variable increases, the other decreases proportionally.
- 0 indicates no relationship between the variables.

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2; \sum(y_i - \bar{y})^2}} \tag{3.1}$$

Where:

- $x_i$ = values of variable $x$ in the sample
- $\bar{x}$ = mean of variable $x$
- $y_i$ = values of variable $y$ in the sample
- $\bar{y}$ = mean of variable $y$

In this work, the absolute value of $r$ was considered, *Figures 3.4*, as it reflects the strength of the correlation independently of its sign. Values closer to 1 indicate stronger linear relationships between the variables.

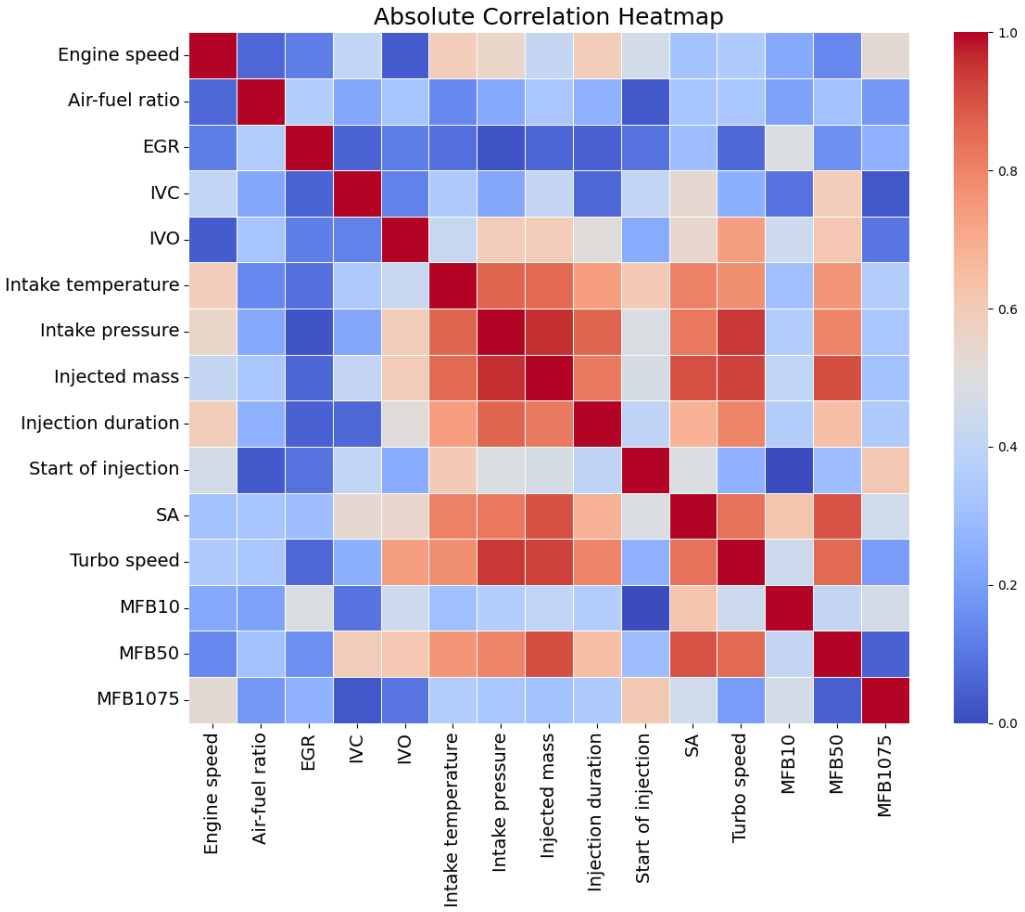



Figure 3.4: Heatmap of Pearson's correlation coefficient

What stands out from this graph is the linear dependency observed between certain features. For example, as expected, the injected mass is highly correlated with the injection duration, while the intake temperature shows a marked correlation with the intake pressure. Another relevant correlation is observable between turbine speed and intake conditions: as the turbine rotates faster, the compressor delivers a higher air mass flow, leading to an increase in both intake pressure and intake temperature.

### 3.1.3 Conclusions

After these purely statistical results, some consideration were made by considering the problem and applying engineering knowledge of the combustion process. It was decided to remove the following inputs features:

- **Intake valve opening** (IVO) and **intake valve closure** (IVC) as they show low correlation with the target outputs, since they were phased to optimize the Miller cycle effect.
- **Injection duration** since it is highly correlated with the injected mass, therefore, it was decided to keep the second one.
- **Start of injection** as it shows low correlation with the combustion targets and primarily affects mixture homogeneity, and thus emissions, rather than the combustion profile.

The final set of 8 selected features is presented in *Table 3.2*.

| **Final Features** | |
|---|---|
| Engine speed | [RPM] |
| Turbo speed | [RPM] |
| Exhaust gas recirculation | [%] |
| Air-fuel ratio -$\lambda$ | [-] |
| Intake pressure | [bar] |
| Intake temperature | [K] |
| Injected fuel mass | [mg/cycle] |
| Spark advance | [deg CA] |

Table 3.2: Final set of features

## 3.2 Probability Density Analysis

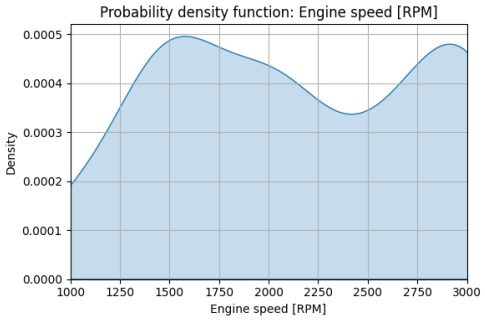
Once the most relevant features were selected, their distribution in the dataset was analyzed to check whether, across the 101 operating points, they were evenly represented or unbalanced. An uneven distribution can reduce the model's performance. The model should be exposed to a wide range of situations during training to improve its predictions in testing. If some feature values are rare and appear

only in the test set, the model may perform poorly because it has never seen those cases before. As a consequence, in the case of an unbalanced distribution, it is important to carefully design the train–test split in order to ensure reliable model evaluation.

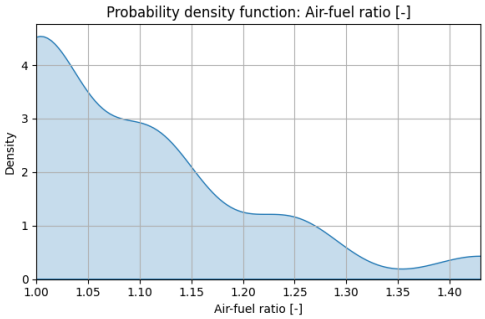
As shown in *Figure 3.5*, the probability density distributions of the selected features highlight several relevant aspects:

- **Engine speed**, *Figure 3.5a*, shows a fairly balanced distribution between 1500 and 3000 RPM, with a slight reduction around 2600 RPM where only a few points are present.
- **Air–fuel ratio ($\lambda$)**, *Figure 3.5b*, is mostly skewed toward stoichiometric or slightly lean mixtures, with very few points in extremely lean conditions.
- **EGR**, *Figure 3.5c*, exhibits a monotonically decreasing trend, with higher probability concentrated at low EGR rates, between 5 and 10%.
- **Turbo speed**, *Figure 3.5h*, and **intake pressure**, *Figure 3.5e*, both characterized by two peaks, reflecting two dominant operating regimes: one at low turbo speed and low intake pressure, corresponding to naturally aspirated or light-load conditions, and another at high turbo speed and high intake pressure, corresponding to boosted high-load operation, with limited occurrence of intermediate states.
- **Load**, *Figure 3.5f*, follows a trimodal distribution, with the engine operating mainly at low, medium, and high load points.
- **Spark advance**, *Figure 3.5g*, shows a two-peak distribution, with two main operating strategies: advanced ignition at light load and retarded ignition at high load.

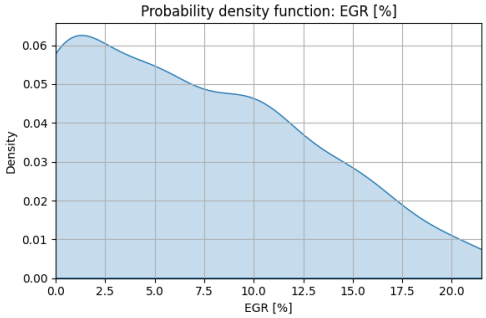
Therefore, it is important that the dataset is split while also considering these distributions, in order to avoid deteriorating the model performance.



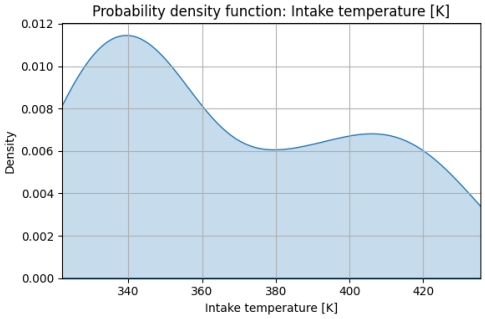

(a) Engine speed



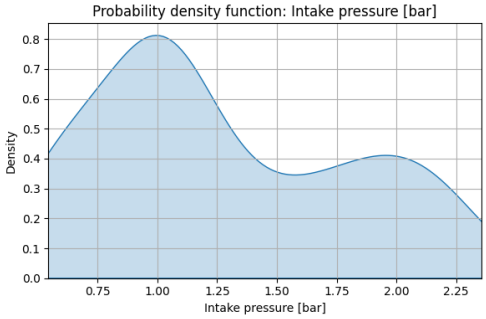

(b) Air–fuel ratio $\lambda$



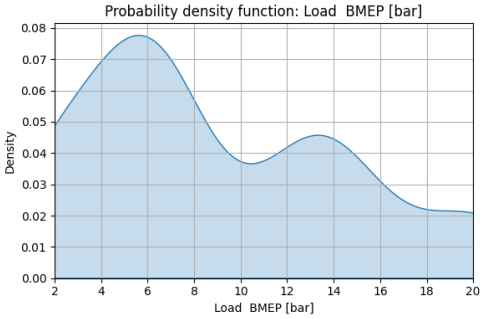

(c) Exhaust Gas Recirculation (EGR)



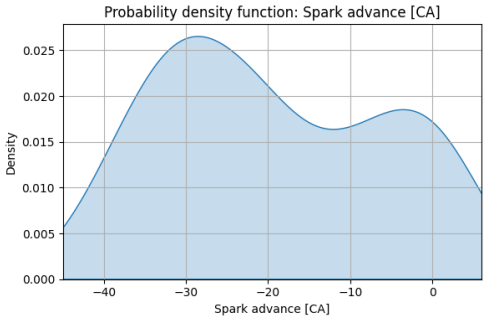

(d) Intake temperature



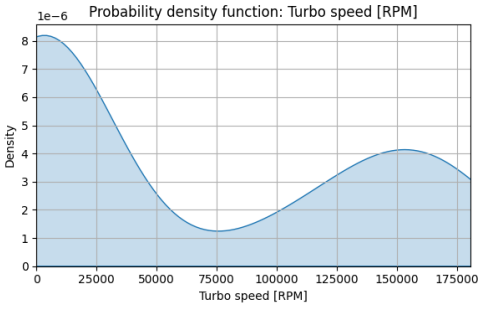

(e) Intake pressure




(f) Load (BMEP)




(g) Spark advance




(h) Turbo speed

Figure 3.5: Probability density analysis of the selected features.

## 3.3 Neural Network for Wiebe Parameter prediction

The first model developed during the dissertation project was a 'hybrid model', a fully connected neural network [35] trained to predict the parameters of the Wiebe function *Eq 3.2*, rather than directly predicting the whole combustion curve. Therefore, the model output a set of parameters that were subsequently used to construct the mass fraction burned (MFB) profile. Afterwards, as a final step, the MFB profile was differentiated with respect to the crank angle to obtain the burn rate profile.

$$\text{Combustion}(\theta) = CE\left[1 - e^{-WC\,(\theta - SOC)^{E+1}}\right] \tag{3.2}$$

Where, $WC$, wiebe constant, and $SOC$, start of combustion, are defined as:

$$WC = \left[\frac{D}{BEC^{\frac{1}{E+1}} - BSC^{\frac{1}{E+1}}}\right]^{-(E+1)} \tag{3.3}$$

$$SOC = AA - \frac{D \cdot BMC^{\frac{1}{E+1}}}{BEC^{\frac{1}{E+1}} - BSC^{\frac{1}{E+1}}} \tag{3.4}$$

In *Eq 3.3* and *Eq 3.4* three additional calculated constants are introduced: $BMC$, $BSC$, and $BEC$, which represent the burned midpoint, start, and end constants, respectively.

- Burned Midpoint Constant: $BMC = -\ln(1 - BM)$
- Burned Start Constant: $BSC = -\ln(1 - BS)$
- Burned End Constant: $BEC = -\ln(1 - BE)$

Where:

- $BM$ = Burned Fuel Percentage at Anchor Angle (50%)
- $BS$ = Burned Fuel Percentage at Start Angle (10%)

- $BE$ = Burned Fuel Percentage at End Angle (90%)

And lastly, the four parameters that the model aims to predict, which vary and must be adapted for each different engine operating condition, are:

- **Anchor Angle** - $AA$ - defined as the crank angle at which 50% of the total mass is burned.
- **Duration** - $D$ - corresponding to the crank angle interval over which the combustion process occurs.
- **Wiebe Exponent** - $E$ - a shape factor that controls the steepness of the mass fraction burned curve.
- **Fraction of Fuel Burned** - $CE$ - representing the effective percentage of fuel burned relative to the total injected mass, accounting for possible incomplete combustion.

The steps followed to create this model were:

- Extraction of the parameters from the experimental curves using genetic algorithms.
- Normalization of both features and targets.
- Definition of the model architecture and selection of the hyperparameters.
- Testing of the model.
- Reconstruction of the combustion curves using the predicted parameters.

### 3.3.1 Genetic Algorithm

The Genetic Algorithm (GA) is an optimization technique based on Darwinian natural selection, which can be applied to both constrained and unconstrained optimization problems. It creates a group of possible solutions, called individuals, and improves them step by step using operations inspired by nature, such as selection, crossover, and mutation, to reach the best solution [8].

*Figure 3.6* illustrates the general workflow of the Genetic Algorithm.

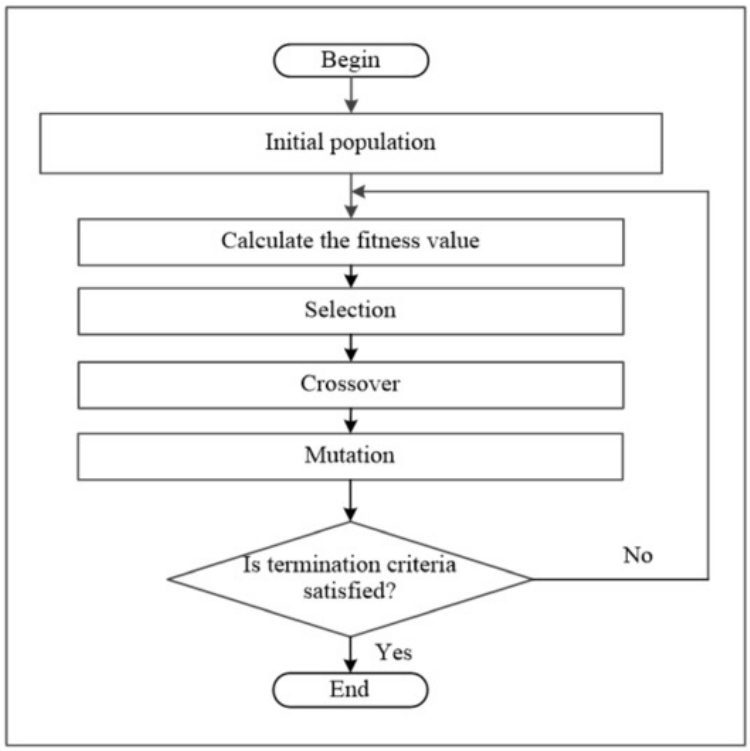



Figure 3.6: Flowchart of the standard genetic algorithm - [8]

The process starts with the generation of an initial population, in this case, a set of possible parameter vectors for the Wiebe function. Each individual in this population is then evaluated by computing a fitness value, which in this case is based on the error between the experimental combustion curve and the curve reconstructed through the Wiebe model.

Once the fitness values are calculated, the algorithm selects some individuals, called parents, to create the next generation. Individuals with higher fitness have a greater chance of being chosen.

The parents are then combined through crossover, where parts of their parameters are mixed to create new children. To preserve diversity and prevent the search from getting stuck too early, mutation is applied, introducing small random changes to some individuals.

Lastly, the algorithm checks if the termination condition is satisfied, such as the saturation of the fitness improvement. If not, the new population is re-evaluated,

and the process repeats. When the condition is met, the algorithm stops and returns the best set of parameters.

The GA was implemented using the Python `PyGAD` library. *Table 3.3* reports the parameters defined for the search of the optimal set of Wiebe parameters.

| **Genetic Algorithm Configuration** | |
|---|---|
| Number of generations | 10000 |
| Parents per generation | 4000 |
| Population size | 9000 |
| Number of genes | 4 |
| Gene space | $AA = [\min(\theta), \max(\theta)]$, $E = [10^{-2}, 50]$, $D = [1, 50]$, $CE = [0.5, 0.99]$ |
| Mutation | 50% of genes, random |
| Crossover | Uniform |
| Stop criterion | Saturation over 50 generations |

Table 3.3: Genetic Algorithm configuration used for Wiebe parameter optimization.

The choices made consists of a relatively large population size and number of parents to ensure a wide search of the solution space, while the mutation rate was set to 50% to preserve diversity and avoid premature convergence. The crossover operator was set to uniform to guarantee an equal probability of exchanging genes between parents. The stopping criterion was set to saturation over 50 generations, ensuring that the algorithm terminates once the improvement in fitness becomes negligible.

The following figures (*Figure 3.7*, *Figure 3.8*, and *Figure 3.9*) compare the experimental MFB and burn rate curves with those obtained from the Wiebe function (*Eq. 3.2*). The function was fitted using the optimal parameters identified through GA optimization.

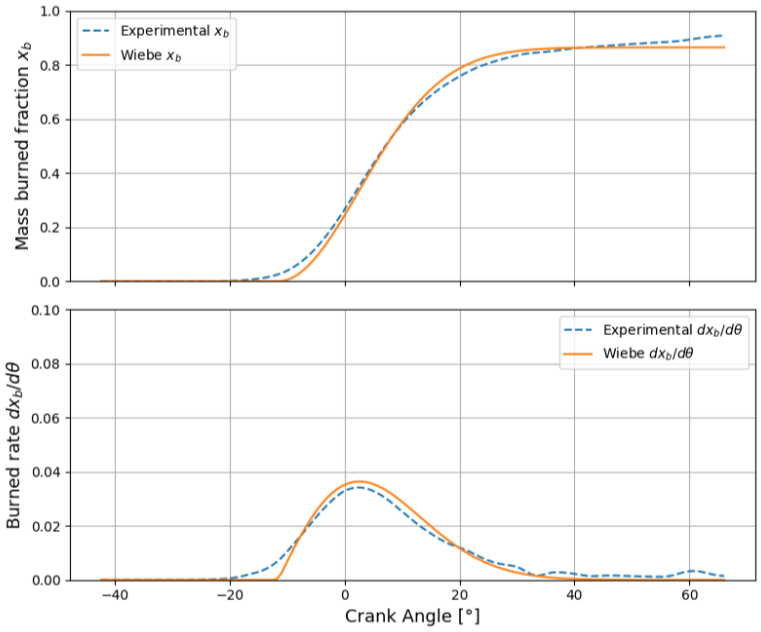



Figure 3.7: Experimental and Wiebe combustion curve - cycle 17

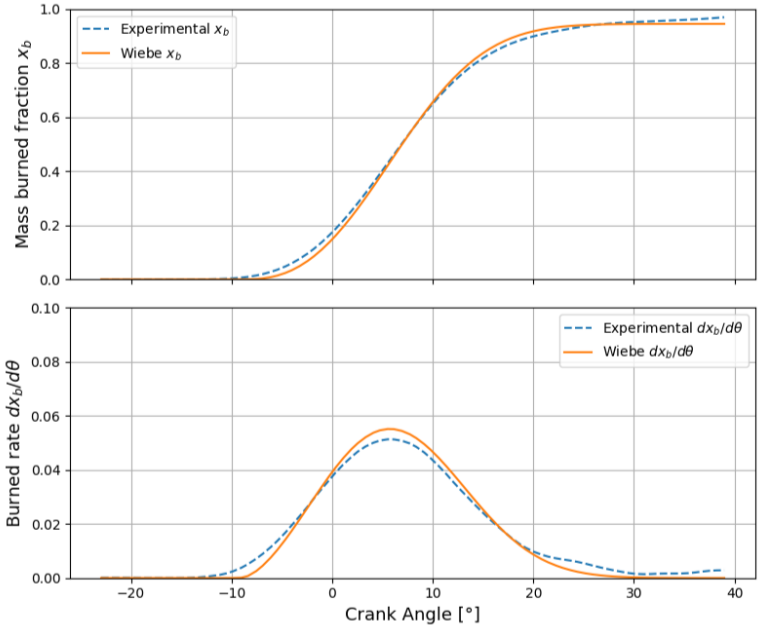



Figure 3.8: Experimental and Wiebe combustion curve - cycle 26

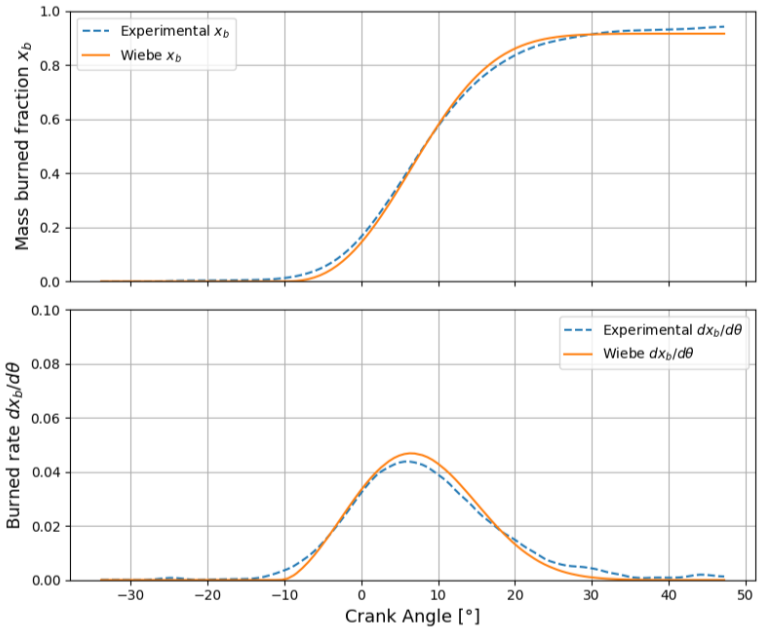



Figure 3.9: Experimental and Wiebe combustion curve - cycle 80

It can be observed that this approach presents some limitations in accurately fitting the start of combustion, particularly in *cycle 17*. Moreover, due to tail noise in the burn rate, the total mass of burned fuel is not fully captured by Wiebe function.

### 3.3.2 Neural network characteristics

In this section, the architecture of the network is presented together with the description of the training and testing procedures

The neural network model was implemented using the Multi-Layer Perceptron Regression (*MLPRegressor*) class from the scikit-learn library, which provides a feedforward artificial neural network trained with backpropagation approach.

#### Neural network architecture

Since the model requires a tuple structure to define the hidden layer configuration, a dedicated function was defined to generate this tuple based on the specified number of layers and neurons for each layer. This approach allowed a flexible definition of

network architecture, which was particularly useful during the optimization process. The implementation is shown in the code below.

```
def fit(self, X, y):
    layers = [self.neurons_layer_1]
    if self.num_layers > 1:
        layers.append(self.neurons_layer_2)
    if self.num_layers > 2:
        layers.append(self.neurons_layer_3)
    if self.num_layers > 3:
        layers.append(self.neurons_layer_4)
    hidden_layer_sizes = tuple(layers)
```

After evaluating different possible architectures, the three top-performing configurations were compared to identify the most suitable for this application. All configurations share the same input and output layer sizes:

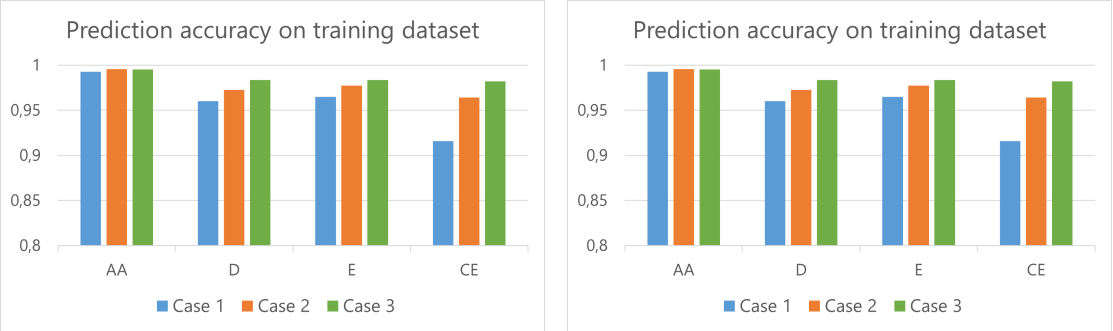
- 8 neurons in the input layer, equal to the number of features selected in *Section 3.1.3*
- 4 neurons in the output layer, equal to the number of Wiebe parameters

The difference lies in the hidden layers:

- **Case 1**: three hidden layers with 16–9–16 neurons
- **Case 2**: three hidden layers with 64–32–16 neurons
- **Case 3**: four hidden layers with 64–32–16–16 neurons

*Figure 3.10* shows the $R^2$ scores obtained when predicting the target parameters on both the training and test sets for the three configurations.

After the network optimization process was completed, *Case 2* was selected. Despite performing slightly worse than *Case 3*, it uses one fewer hidden layer, which reduces model complexity and the risk of overfitting, improving generalization on unseen data.




**(a)** Training dataset **(b)** Testing dataset

Figure 3.10: $R^2$ scores of the predicted parameters for three network configurations

### Model hyperparameters

The model hyperparameters were selected as follows:

- **Activation function**: the *tanh* (hyperbolic tangent) function was adopted, as it effectively captures nonlinear relationships and provides more stable gradients compared to the sigmoid function.
- **Optimizer**: the Adam algorithm was used, since it combines the advantages of adaptive learning rates with momentum, ensuring both robustness and efficiency.
- **Initial learning rate**: set to $1 \times 10^{-2}$, which allows relatively fast convergence, especially during the early training phase.
- **Regularization (L2)**: the value $\alpha = 1 \times 10^{-3}$ was chosen to penalize excessively large weights and reduce the risk of overfitting.
- **Stopping criteria**: the training process was limited to a maximum of 8000 iterations, with a tolerance of $1 \times 10^{-7}$, and included early stopping after 50 iterations without improvement.

This configuration was selected after preliminary testing of different architectures and hyperparameter ranges, balancing convergence speed, model complexity, and generalization performance.

### 3.3.3 Model fitting and testing

Model fitting and model testing are two fundamental processes for achieving a good match between predicted and observed data. The subset used to train the model must be different from the one used for testing, so that the evaluation reflects the model's ability to generalize to unseen data.

In this study, an 80/20 split was adopted: 80% of the data were used to train the model and 20% were adopted for testing. Specifically, among the 101 operating points, 80 were used for training and 21 for testing.

## 3.4 Gaussian Process Regression (GPR) for burn rate prediction

The second model developed and analyzed was the Gaussian Process Regression (GPR). Unlike neural networks, GPR is a non-parametric and probabilistic approach, which means it does not rely on a predefined structure to model the relationship between inputs and outputs, but instead learns it directly from the data. As a consequence, GPR requires fewer design choices from the user, relying mainly on kernel selection. An interesting feature of this model is that it provides predictions together with a probability distribution, allowing an estimation of the associated uncertainty.

The following development steps were carried out to design the model and identify the most suitable configuration:

- Processing of the dataset to impose correct input and output targets
- Normalization of the dataset, including both input and output variables
- Definition and evaluation of kernel functions, together with the data splitting strategy

### 3.4.1 Dataset pre-processing

The inputs and outputs of this model differ from those used in the previous one. In this case, the objective was to directly predict the burn rate curve without relying

on the Wiebe function. Subsequently, the inputs consisted of the eight features selected in *Section 3.1.3*, repeated across the crank angle domain and combined with the crank angle position, while the target output was the burn rate value at each corresponding crank angle under the given operating conditions. This mapping can be formally expressed as:

$$X = \Big[\text{features of the cycle, } \theta\Big] \quad \longrightarrow \quad y = \text{burn rate at } \theta \tag{3.5}$$

### 3.4.2 Dataset splitting strategy

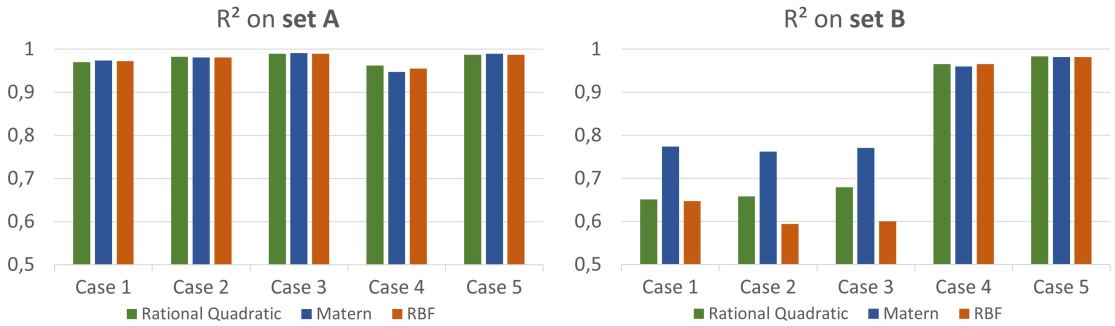
The dataset was divided into two subsets, each containing the filtered combustion sequences, to mitigate the noise present in the combustion tail, together with their corresponding operating features (engine speed, EGR percentage, etc.). The larger subset, consisting of 81 sequences, is referred to as **Set A**, while the smaller subset, containing 20 sequences, is referred to as **Set B**.

The performance of the GPR model, with the three kernel functions defined previously, was then evaluated under five different training configurations:

- **Case 1:** 1,620 points were randomly selected from Set A to fit the model.
- **Case 2:** 20 points were randomly selected from each sequence in Set A to fit the model.
- **Case 3:** 20 evenly spaced points were selected from each sequence in Set A to fit the model.
- **Case 4:** 1,620 points were randomly selected from the entire dataset (Set A + Set B) to fit the model.
- **Case 5:** 4,000 points were randomly selected from the entire dataset (Set A + Set B) to fit the model.

The predictive accuracy across these cases was compared using two performance metrics: the Root Mean Squared Error (RMSE) and the coefficient of determination ($R^2$). In *Figure 3.11*, the $R^2$ score is used to present the results.

Different behaviors can be observed between *Set A* and *Set B*. In particular, the accuracy of the predictions for the larger dataset (Set A) remains consistently




**(a)** Subset A **(b)** Subset B

Figure 3.11: $R^2$ scores of the predictions obtained with different kernels and training methodologies

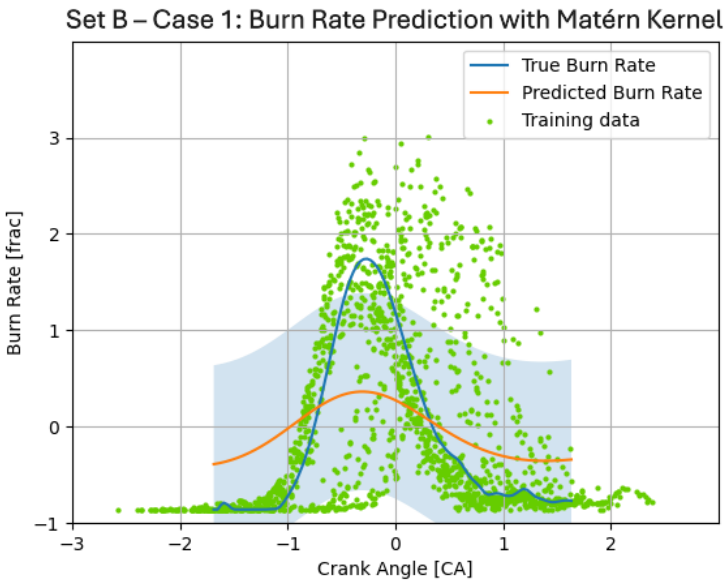
above 0.95, with no substantial differences among the kernel functions. By contrast, the first three cases of Set B exhibit lower performance, with accuracy values below 0.80; under these conditions, the Matérn kernel provides slightly better results than the other kernels. However, the last two cases of Set B show performance comparable to that of Set A.

These observations allow us to draw some preliminary conclusions about the model. First, as shown in the first three cases, the model is not capable of extrapolating new functions to fit unseen data. On the contrary, even with a very limited number of sequence points, the model is still able to interpolate the sequence effectively and provide a reliable prediction of the curve, as demonstrated in the last two cases, *Case 4* and *Case 5*.

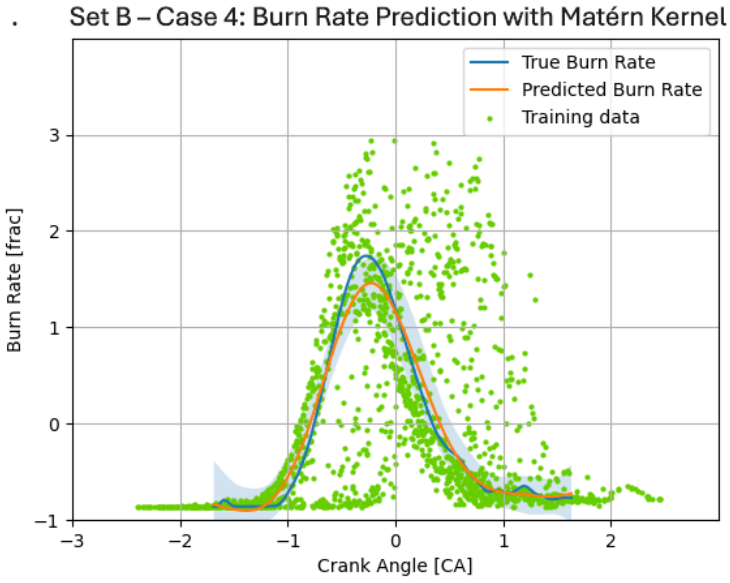
In *Figure 3.12*, a comparison is reported between the experimental curve, the predicted one, and the distribution of the data used to train the model in two different cases, *Case 1* and *Case 4*. Both cases are characterized by the same number of fitting points (1620), but differ in the data splitting strategy.

Despite the comparable distribution of training data (green dots), since the number of points is the same, a clear difference in accuracy can be observed between *Figure 3.12a* and *Figure 3.12b*, with the latter outperforming the former. This confirms what was stated before, in particular that the model is able to interpolate the data better than to extrapolate it.

The final configuration adopted for presenting the results in the last chapter




(a) Case 1




(b) Case 4

Figure 3.12: Comparison between predicted and experimental burn rate using the Matérn kernel.

is a Gaussian Process Regression (GPR) model with a Matérn kernel, trained on *Case 5*. The model was optimized with respect to the hyperparameter *length_scale* within the range $(10^{-3}, 10^{3})$, while the parameter $\nu$ was fixed at 1.5.

## 3.5 Neural Network for burn rate prediction

The third and last model presented in this dissertation is a fully connected Neural Network (NN) designed to directly predict the combustion profile, rather than the Wiebe parameters.

The workflow adopted for the development of this model is a combination of steps from the procedures applied in the first and second models:

- Similarly to the GPR model, the dataset was pre-processed to obtain the desired sequence of inputs and outputs.
- As in the first model, the dataset was split using the conventional 80/20 technique.
- The model architecture was defined, and the hyperparameters were optimized.
- The model was then tested and evaluated.

### 3.5.1 Dataset pre-processing

Similarly to the GPR model, the inputs consisted of the eight features selected in *Section 3.1.3*, repeated across the crank angle domain and combined with the crank angle position, while the target output was the burn rate at each corresponding crank angle under the given operating conditions. Unlike the GPR case, an equal vector length, for all the 101 operating points, was imposed to facilitate the capture of trends.

### 3.5.2 Dataset splitting strategy

In this study, an initial 80/20 split was adopted: 80% of the data were used to train the model and 20% were adopted for testing. Specifically, among the 101 operating points, 80 were used for training and 21 for testing.

Once the model hyperparameters were optimized, a different data-splitting strategy was adopted to evaluate the robustness of the model under more challenging conditions. In particular, the model was trained on a dataset excluding all the sweeps performed at 1500 RPM and 5.5 bar BMEP, in order to assess its performance on completely unseen data.

### 3.5.3 Neural network characteristics

In this section, the main characteristics of the neural network are presented.

The neural network model was implemented using the Multi-Layer Perceptron Regression (*MLPRegressor*) class from the scikit-learn library, as in *Section 3.3*

#### Model hyperparameters

During the definition of the hyperparameters, some were selected directly, while others were chosen through an optimization process. The ones set beforehand are:

- **Optimizer**: the Adam algorithm was selected.
- **Initial learning rate**: set to $1 \times 10^{-2}$.
- **Stopping criteria**: the training process was limited to a maximum of 8000 iterations, with a tolerance of $1 \times 10^{-2}$, and included early stopping after 100 iterations without improvement, in order to reduce the risk of overfitting.

Instead the ones selected after an optimization are:

- **Activation function**: three different activation functions were tested:
  - Hyperbolic tangent (tanh)
  - ReLU
  - Logistic
- **Regularization (L2)**: three different values were tested:
  - $\alpha = 1 \times 10^{-1}$
  - $\alpha = 1 \times 10^{-2}$

- $\alpha = 1 \times 10^{-4}$

Firstly, a sensitivity analysis was conducted by varying the regularization factor and the activation function. Once the former was selected, a second analysis on network dimensionality and activation function was carried out to obtain the final configuration.

In *Figure 3.13*, the variation of RMSE is presented as a function of the activation function and the regularization factor, evaluated with respect to the three main combustion parameters: MFB10, MFB50, and MFB1075. The results are shown for both the training and the test datasets, emphasizing the importance of correctly selecting the regularization factor to avoid overfitting and underfitting.

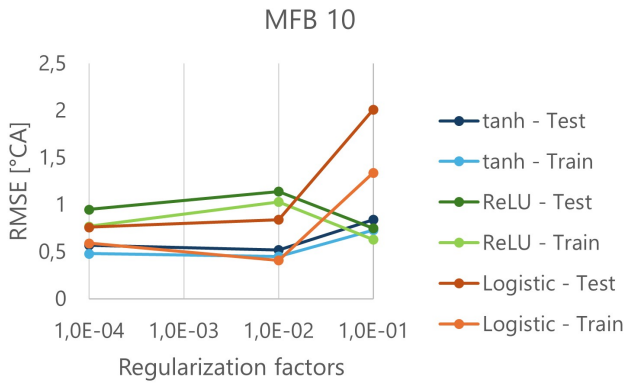
Firstly, it can be observed that, overall, the RMSE values of the training dataset are consistently lower than those of the test dataset across all three graphs. However, the errors between the two subsets are comparable, meaning that the model does not show problems of overfitting or underfitting, but is able to generalize.

Starting from *Figure 3.13a*, the regularization factor $\alpha = 1 \times 10^{-4}$ ensures more consistent performance across the different activation functions and subsets. However, also for $\alpha = 1 \times 10^{-2}$, a negligible variation in RMSE between the training and test sets can be observed, especially with the ReLU and tanh activation functions. In contrast, with the logistic function, the performance of the two sets starts to diverge. Lastly, for $\alpha = 1 \times 10^{-1}$, the performance with the logistic function worsens dramatically compared to the other cases; with the hyperbolic tangent it also deteriorates slightly, whereas for ReLU the trend is inverted, leading to improved performance. A similar trend can be observed for MFB50, as shown in *Figure 3.13a*.

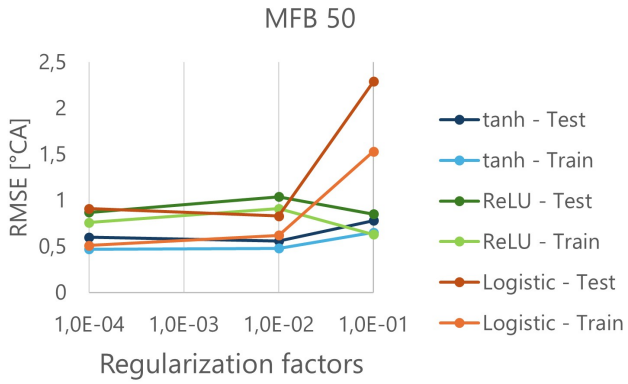
Lastly, *Figure 3.13c* shows slightly different trends. In fact, the best results are obtained for $\alpha = 1 \times 10^{-2}$ in most cases, while the performance slightly worsens for $\alpha = 1 \times 10^{-4}$, though less than for $\alpha = 1 \times 10^{-1}$.

Based on this analysis, the regularization factor $\alpha = 1 \times 10^{-2}$ was selected because it guarantees reliable performance across different cases without excessively penalizing large weights.

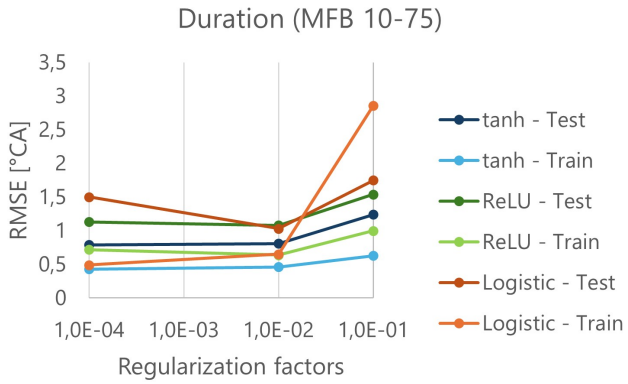
After that, the focus was shifted to the study of the network dimensionality.




(a) RMSE of MFB10




(b) RMSE of MFB50




(c) RMSE of duration (MFB 10-75)

Figure 3.13: Sensitivity analysis of regularization factor and activation function

### Neural network architecture

The model architecture was optimized to ensure consistent performance in both the training and testing phases.

Specifically, the neurons' number of input and output layers is fixed, because it is imposed by dimensionality of the input features and the target to be predicted. What can be tuned, and what mainly influences the model's performance, is the design of the hidden layers, whose structure plays a decisive role as they concentrate the computational workload.

Therefore, as previously stated, all configurations share the same input and output layer sizes.

- 9 neurons in the input layer, equal to the number of features selected in *Section 3.1.3* with the addition of crank angel:
  $X = \left[\text{features of the cycle, } \theta\right]$
- 1 neurons in the output layer, which correspond to the burn rate to the corresponding crank angle $\theta$

Conversely, several hidden layer configurations were tested in order to identify the one that best mitigates both overfitting and underfitting; some of these are reported and compared with each other.

- **Case 1**: two hidden layers with 8-4 neurons
- **Case 2**: two hidden layers with 16-8 neurons
- **Case 3**: two hidden layers with 32–16 neurons
- **Case 4**: three hidden layers with 16-8-4 neurons

A parallel sensitivity analysis of the network dimensionality and activation function was carried out to obtain the final configuration.

*Figure 3.14* shows the variations in the RMSE of the three main combustion parameters: MFB10, MFB50, and MFB1075.

Also in this case, it can be observed that the RMSE values of the training dataset are consistently lower than those of the test dataset, as was the case in the previous analysis.

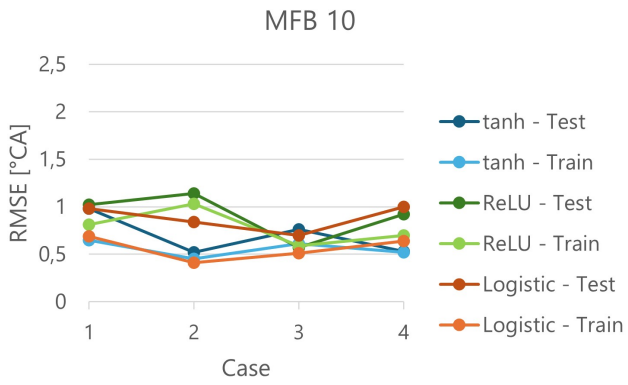
From *Figure 3.14a* and *Figure 3.14b*, it can be observed that *Case 3* shows superior performance compared to the other configurations. When considering the RMSE of the duration prediction (*Figure 3.14c*), both *Case 2* and *Case 3* exhibit similar performance across most activation functions.

Regarding the selection of the activation function, both ReLU and *tanh* showed similar performance in *Case 3*; however, ReLU was ultimately chosen due to its lower computational complexity and higher efficiency.

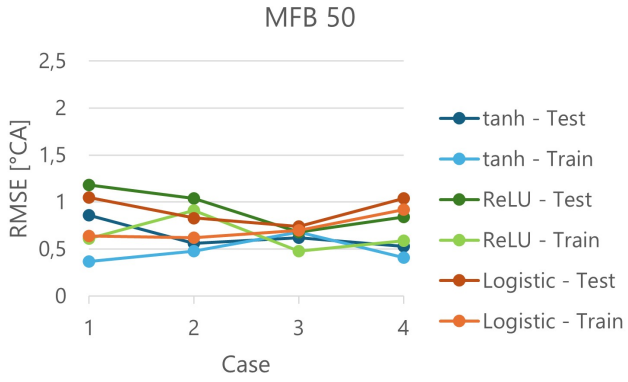
The final configuration of the neural network is summarized in *Table 3.4*

| **Hyperparameter** | **Value** |
|---|---|
| Hidden layers | [32, 16] |
| Regularization ($\alpha$) | $1 \times 10^{-2}$ |
| Activation function | ReLU |
| Optimizer | Adam |
| Initial learning rate | $1 \times 10^{-2}$ |
| Stopping criteria | Maximum of 8000 iterations, tolerance of $1 \times 10^{-2}$, early stopping after 100 iterations without improvement |

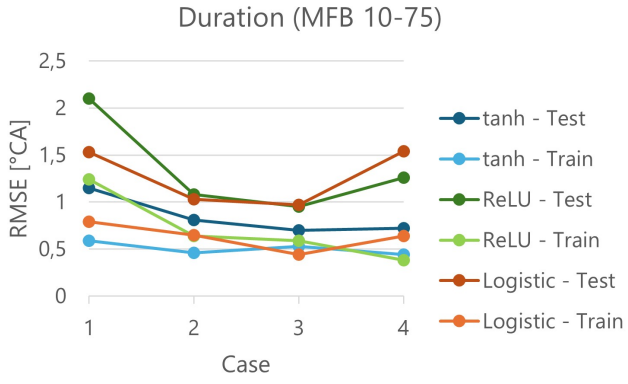
Table 3.4: Final configuration of the neural network model.




(a) RMSE of MFB10




(b) RMSE of MFB50




(c) RMSE of duration (MFB 10-75)

Figure 3.14: Sensitivity analysis of hidden layer dimensionality and activation function

# Chapter 4

# Results

In this section, the results obtained from the simulations are presented and discussed. The analysis focuses on evaluating the performance of the models under different sweeps of lambda ($\lambda$) and EGR, mainly for two different engine operating points, 1500 RPM $\times$ 5.5 bar BMEP and 3000 RPM $\times$ 7.0 bar BMEP. The discussion will highlight the main trends that emerged from the analyzed data. Particular attention is given to the comparison between predicted and experimental burn rate curves, as well as to the main combustion parameters.

In *Figure 4.1* the structure of the analyzed combinations of lambda and EGR for the results is reported

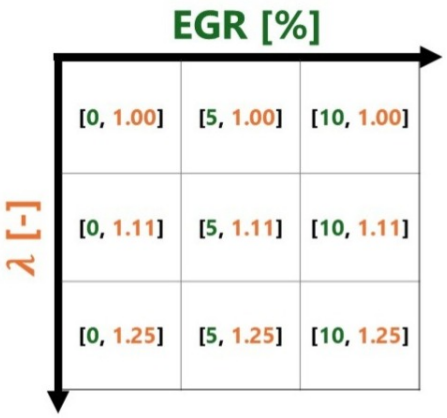



Figure 4.1: Burn rate curves - structure sweep of $\lambda$ and EGR

Instead, *Figure 4.2* shows the sweep structure in which the predicted combustion metrics are analyzed and compared with the experimental ones.

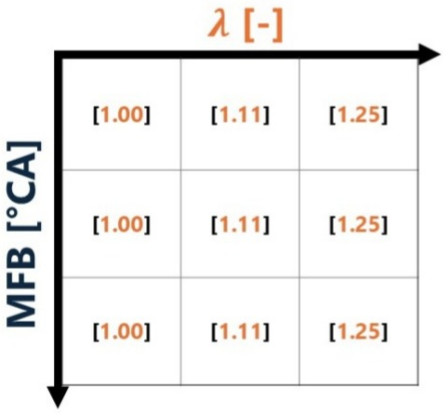


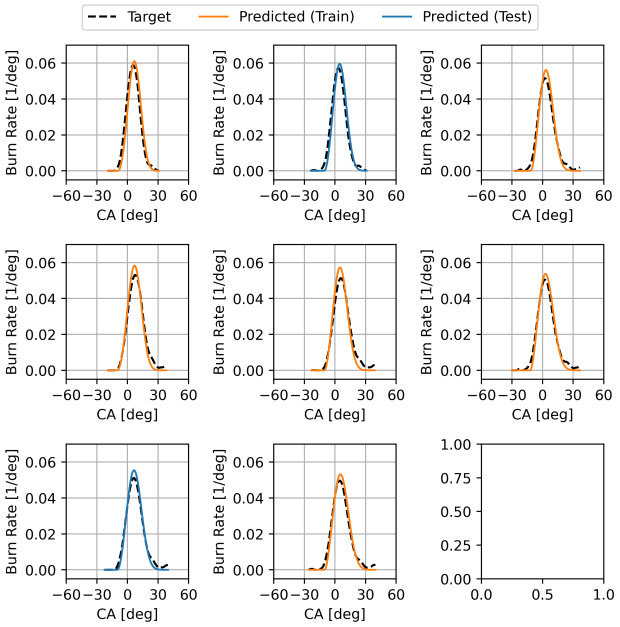

Figure 4.2: Combustion metrics prediction - structure sweep of $\lambda$

## 4.1 Neural Network for Wiebe parameters prediction

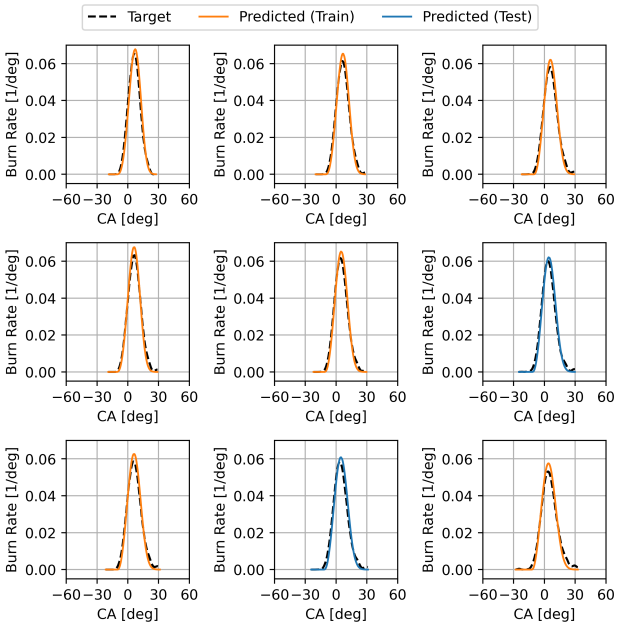
In *Figure 4.3a* and *Figure 4.3b*, the comparison between experimental and predicted curves for the sweeps of $\lambda$ and EGR is reported, following the structure of *Figure 4.1*. Overall, the model is able to capture the burn rate profile, but with some limitations in accuracy. In particular, for both operating conditions, 1500 RPM$\times$5.5 bar BMEP and 3000 RPM $\times$ 7.0 bar BMEP, it tends to overestimate the burn rate peak and shows some difficulties in capturing the start and the end of combustion.

Furthermore, in *Figure 4.4a* and *Figure 4.4b*, a more detailed analysis of the fundamental combustion metrics, such as MFB10, MFB50, and the duration MFB1075, is reported. Starting with MFB10, as mentioned before, the model generally overestimates the start of combustion by a few crank angle degrees. On the other hand, the center of combustion is well predicted in almost all conditions, although with slightly lower precision in the test predictions. Lastly, the duration is consistently underestimated by the network.

The trends highlighted in *Figure 4.4a* and *Figure 4.4b* for the two engine

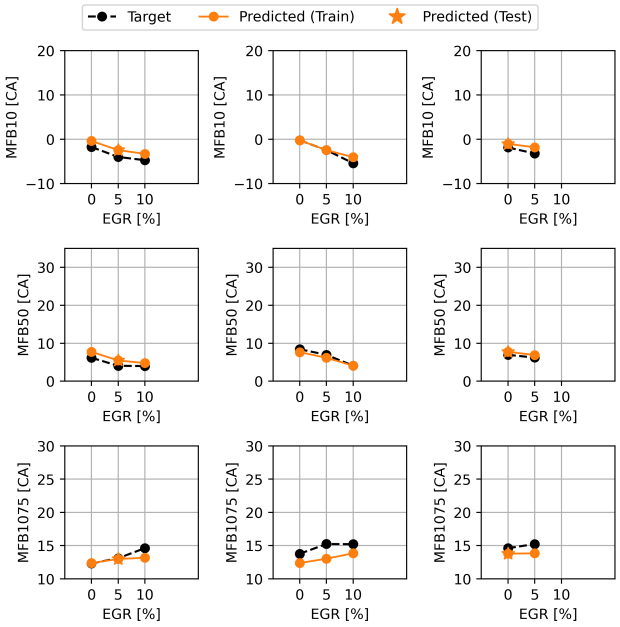



(a) Sweep of $\lambda$-EGR at 1500 RPM $\times$ 5.5 bar BMEP

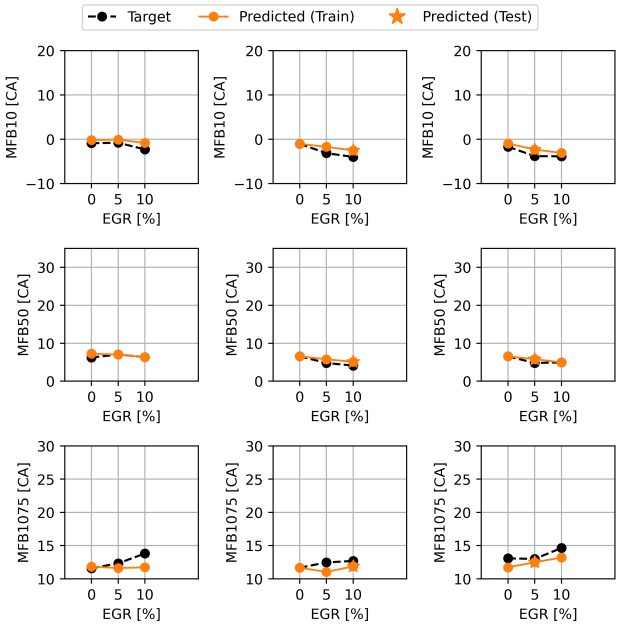



(b) Sweep of $\lambda$-EGR at 3000 RPM $\times$ 7 bar BMEP

Figure 4.3: Burn rate curve predictions using the Neural Network for Wiebe parameters across two operating conditions.




(a) Sweep of $\lambda$ at 1500 RPM $\times$ 5.5 bar BMEP




(b) Sweep of $\lambda$ at 3000 RPM $\times$ 7 bar BMEP

Figure 4.4: Combustion metrics prediction using the Neural Network for Wiebe parameters prediction.

operating conditions, 1500 RPM × 5.5 bar BMEP and 3000 RPM × 7.0 bar BMEP, are confirmed to hold across all conditions, as shown in *Figure 4.5*. An overall overestimation of the start of combustion, a more accurate prediction of MFB50, and a general underestimation of the duration can be observed, although all points remain within the error band of ±5 crank angle degrees.

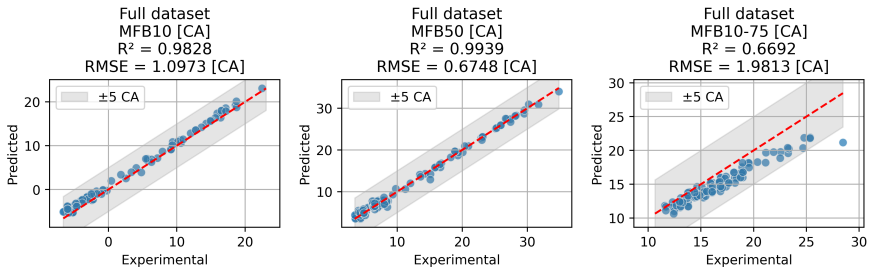



Figure 4.5: Correlation plots of combustion metrics - full dataset - Neural Network for Wiebe parameters prediction

It is worth mentioning that these errors between the experimental and predicted values mainly arise from the imperfect initial fitting of the experimental curves by the genetic algorithm, as explained in *Section 3.3.1*. In fact, when the correlation plots of the combustion metrics are generated between the curves obtained with the genetic algorithm and the predicted ones, the results show an improvement, as reported in *Figure 4.6*. These improvements are mainly associated with the prediction of the initial phase of combustion and with the overall duration.

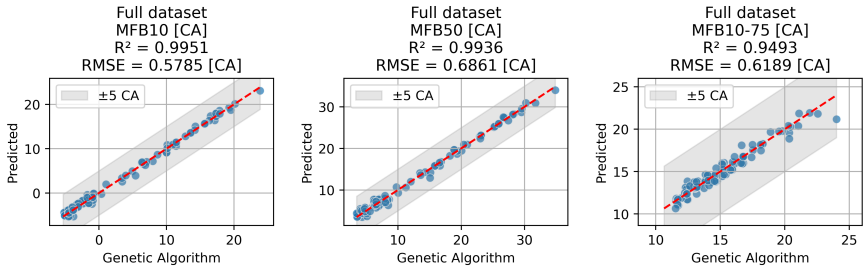



Figure 4.6: Correlation plots of combustion metrics - full dataset - GA-fitted curves and model predictions

### 4.1.1 Conclusions

From this analysis, it is possible to conclude that the Neural Network model is able to accurately predict the Wiebe parameters extracted through the Genetic Algorithm, as shown in the correlation plot of *Figure 4.6*. However, due to the limited accuracy of the algorithm in fitting the burn rate under these ultra-lean and challenging conditions, the predictions compared to the experimental results show lower accuracy, as reported in *Figure 4.5*.

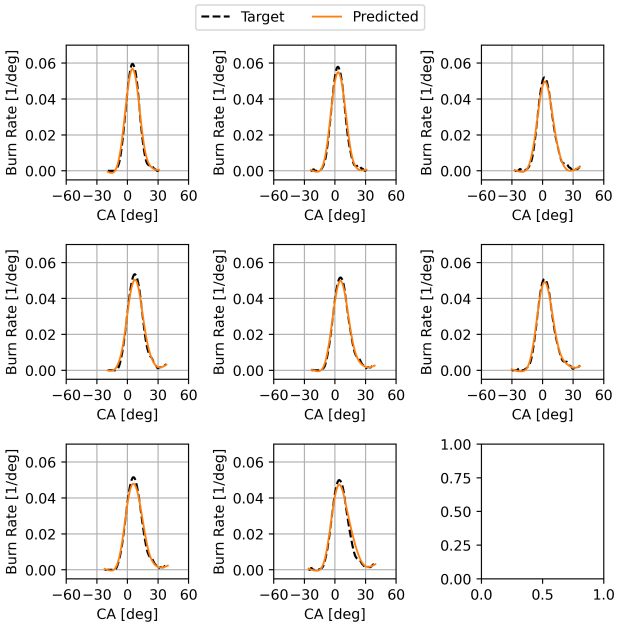
## 4.2 Gaussian Process Regression for burn rate prediction

Furthermore, the results analysis is extended by examining the accuracy achieved by the Gaussian Process Regression model.

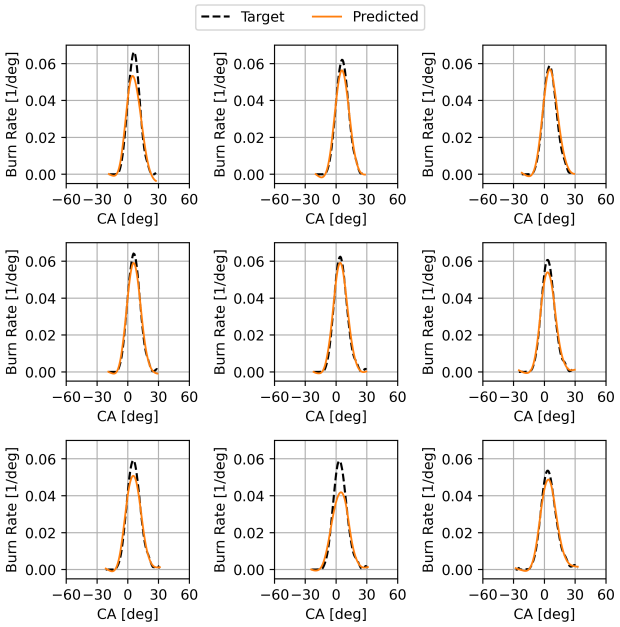
The GPR model differs from the Neural Network model because the dataset is not split between training and testing. Instead, it is fitted by randomly selecting a few points from the entire dataset, about 4,000 points out of more than 60,000 in this case.

Despite the limited number of points used to fit the model (less than 7%), good prediction accuracy can be observed for both operating points, as reported in *Figure 4.7*. A slight underestimation of the burn rate peak is visible, except for two cases at 3000 RPM $\times$ 7.0 bar BMEP where this effect is more pronounced, and as mentioned in the model description, in those cases the model also returns a high uncertainty in the prediction. At the same time, the model shows a higher capability in capturing the start of combustion and the first part of the process, as well as the tail of the profile.

Furthermore, the observations from *Figure 4.7* are confirmed by the analysis of *Figure 4.8*. The model is able to correctly predict both MFB10 and MFB50 for all the reported cycles with very good accuracy. On the other hand, the duration of the predicted curve is slightly longer, in terms of crank angle, than the experimental one. This occurs especially for the 1st point ($\lambda$ = 1 - EGR = 0%) and the 6th point ($\lambda$ = 1.25 - EGR = 5%) in *Figure 4.8b*, due to the lower peak in the burn rate curves.

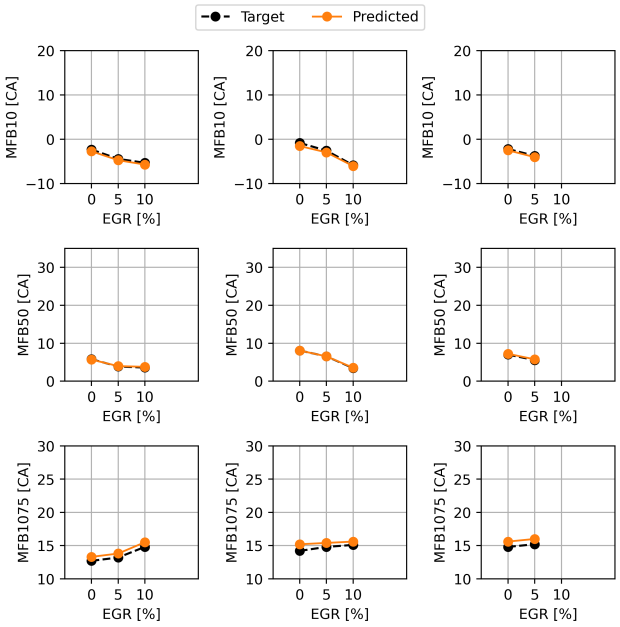



(a) Sweep of $\lambda$-EGR at 1500 RPM $\times$ 5.5 bar BMEP

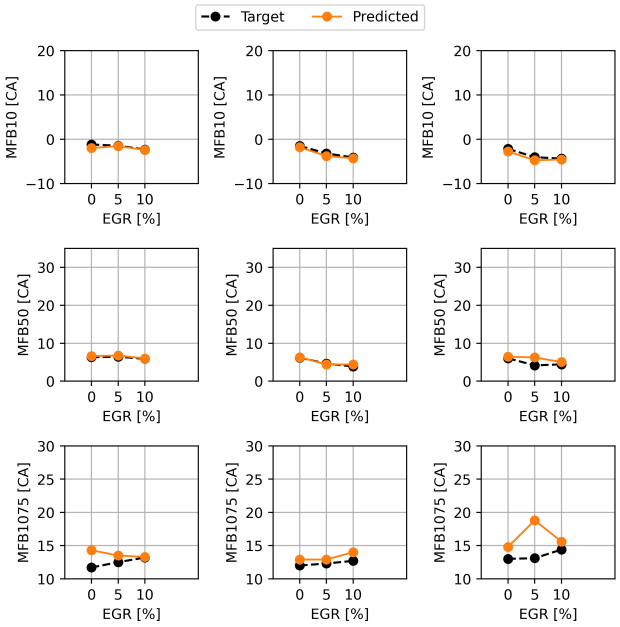



(b) Sweep of $\lambda$-EGR at 3000 RPM $\times$ 7 bar BMEP

Figure 4.7: Burn rate curve predictions using the Gaussian Process Regression




(a) Sweep of $\lambda$ at 1500 RPM $\times$ 5.5 bar BMEP




(b) Sweep of $\lambda$ at 3000 RPM $\times$ 7 bar BMEP

Figure 4.8: Combustion metrics prediction using the Gaussian Process Regression

A general improvement in accuracy compared to the first model is noticeable from the trend of the correlation plots in *Figure 4.9*, as well as from the RMSE values reported above them. From the first Neural Network model to the GPR model, the following reductions in RMSE [°CA] were obtained: 63%, 51%, and 55% for MFB10, MFB50, and duration (MFB1075), respectively. Nevertheless, a general overestimation of the duration remains visible in the last graph of *Figure 4.9*.

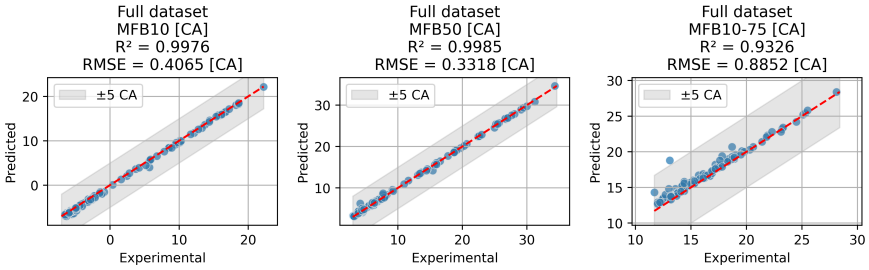



Figure 4.9: Correlation plots of combustion metrics - full dataset - GPR prediction

### 4.2.1 Conclusions

The analysis carried out allows us to state that the Gaussian Process Regression model achieves high prediction accuracy, especially for combustion metrics such as MFB10 and MFB50, since all the points in the correlation plots are very close to the perfect correlation line (the red line in *Figure 4.9*). An overall improvement is also visible in *Figure 4.7*, where the experimental and predicted curves show greater overlap.

## 4.3 Neural Network for burn rate prediction

As mentioned in *Section 3.5.2*, the Neural Network model trained to directly predict the burn rate curves, rather than the Wiebe parameters, was tested under two different conditions, which differ only in the way the full dataset was divided.
The two cases are:

- **Case 1**: the dataset is divided randomly using the 80/20 technique.

- **Case 2**: the dataset was divided by assigning all operating sweeps of 1500 RPM× 5.5 bar BMEP to the test set, while the remaining points were used to train the model.

### 4.3.1 Case 1

The model in the first scenario is able to generalize the predictions well. As shown in *Figure 4.10*, for both testing and training points the network fits the experimental burn rate with high accuracy. It captures the peaks as effectively as the Wiebe-based model. However, unlike the first Neural Network model, this approach also achieves high accuracy in predicting the start and end of combustion, which were the most challenging aspects for the Wiebe parameter prediction model.

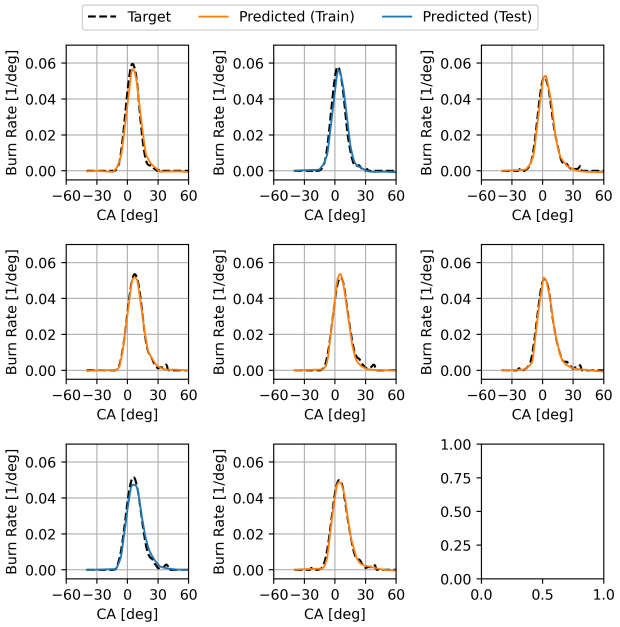
Based on the analysis of *Figure 4.11*, it can be observed that, as mentioned before, the model is able to capture almost all the MFB10 values across the sweeps and for both engine operating points, 1500 RPM × 5.5 bar BMEP and 3000 RPM × 7.0 bar BMEP, as well as MFB50. Additionally, it outperforms all the other models in predicting the duration (MFB1075), since in the last graphs of both *Figure 4.11a* and *Figure 4.11b* there is a good overlap between the experimental and predicted values.

This good performance can be attributed to the overall dataset, as shown in *Figure 4.11a*. Indeed, the $R^2$ values for MFB10 and MFB50 are both higher than 0.994 over the entire dataset, with improvements of 1.17% and 0.14%, respectively, compared to the Wiebe-based model. The largest improvement in accuracy occurs for the duration, with an increase of about 39%.

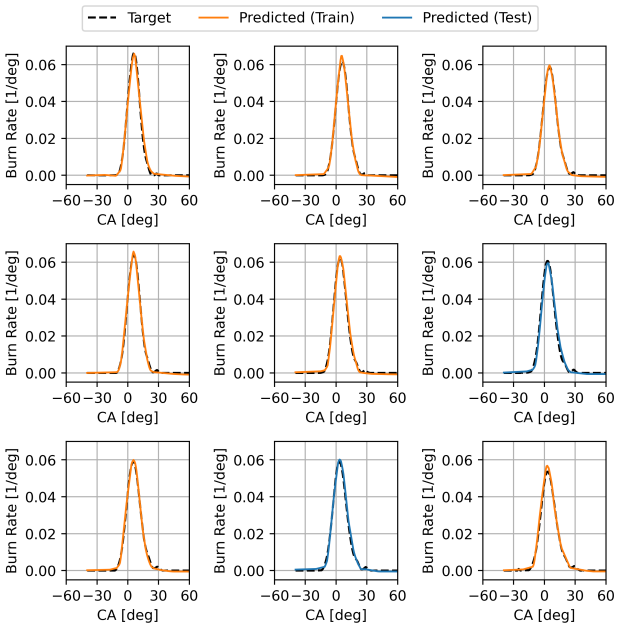
On the other hand, when compared with the results of GPR, only a very limited deterioration can be observed. This drawback can be compensated by the reduced computational cost of the model and its ability to extrapolate more effectively on unseen data, as will be shown in the second case.

### 4.3.2 Case 2

This case study was carried out to evaluate how the model reacts when predicting completely new data. As already stated, the model was retrained by removing all the sweeps at 1500 RPM × 5.5 bar BMEP from the dataset, and the following

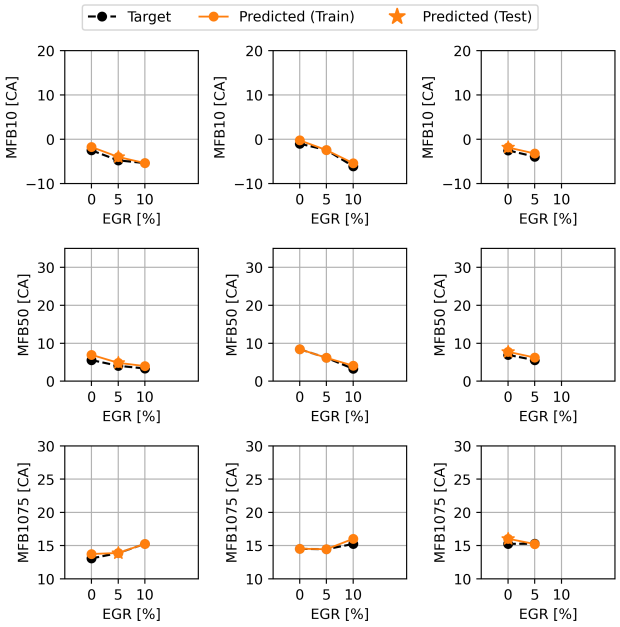



(a) Sweep of $\lambda$-EGR at 1500 RPM $\times$ 5.5 bar BMEP

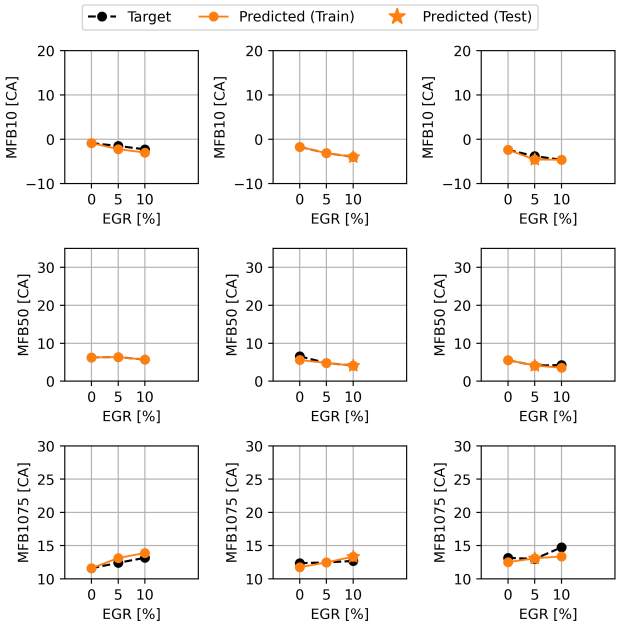



(b) Sweep of $\lambda$-EGR at 3000 RPM $\times$ 7 bar BMEP

Figure 4.10: Burn rate curve predictions using the Neural Network for burn rate prediction - Case 1




(a) Sweep of $\lambda$ at 1500 RPM $\times$ 5.5 bar BMEP




(b) Sweep of $\lambda$ at 3000 RPM $\times$ 7 bar BMEP

Figure 4.11: Combustion metrics prediction using the Neural Network for burn rate prediction - Case 1

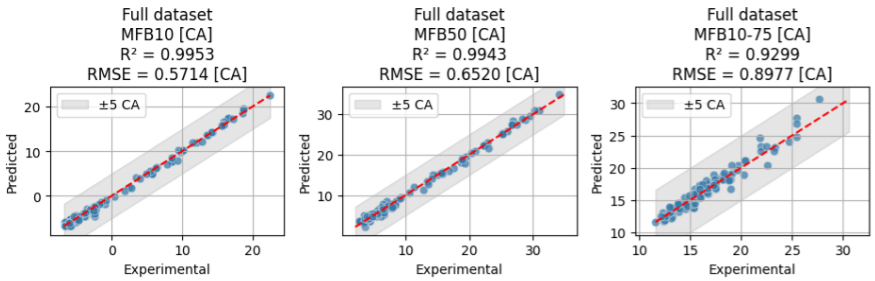



Figure 4.12: Correlation plots of combustion metrics - full dataset - neural network for burn rate prediction - Case 1

results emerged.

Considering that the performance over the training data remains almost constant, this analysis will focus only on the prediction of unseen data.

Starting from *Figure 4.13*, it can be observed that the accuracy of the predicted curves degrades; however, the general trend is still captured by the model. In most of the plots, it is noticeable that the curves are shifted to the right, meaning that the model mispredicts the start of combustion, while the peaks are also slightly underestimated.

The shift phenomenon is clearly highlighted by the MFB50 plots in *Figure 4.14*, where all the predicted points are consistently above the experimental ones, indicating that the combustion is shifted to the right. Additionally, from the latest plots where the duration is reported, it can be observed that it is generally overestimated.

### 4.3.3 Conclusions

It is possible to conclude that this Neural Network model for burn rate prediction is overall the best-performing one, considering accuracy, computational time, and the ability to generalize on unseen data. In fact, with a relatively small network - 1 input, 1 output, and 2 hidden layers with 32 and 16 neurons - this model is able to generalize well between test and training data, while maintaining a relatively limited computational cost compared to the GPR. Lastly, the limited performance on unseen data may also be due to the small number of total points (101), where removing an entire sweep can be excessive. Therefore, future tests with a larger dataset could be carried out, and the performance may be improved.

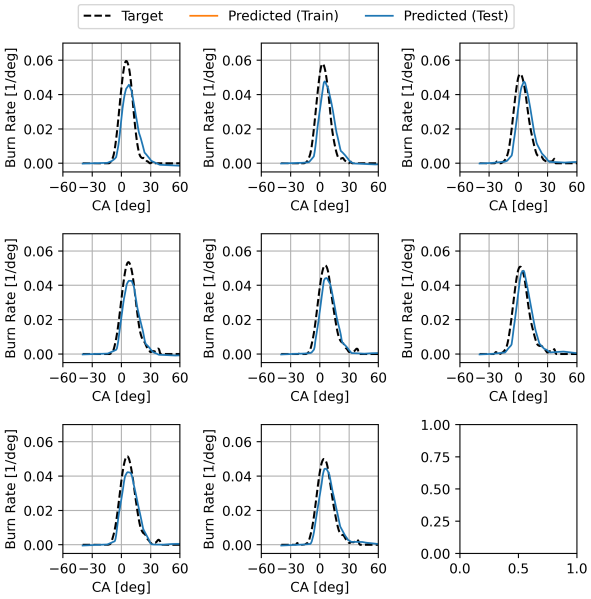



Figure 4.13: Burn rate curves - sweep $\lambda$-EGR at 1500 RPM $\times$ 5.5 bar BMEP - Neural Network for burn rate prediction - Case 2

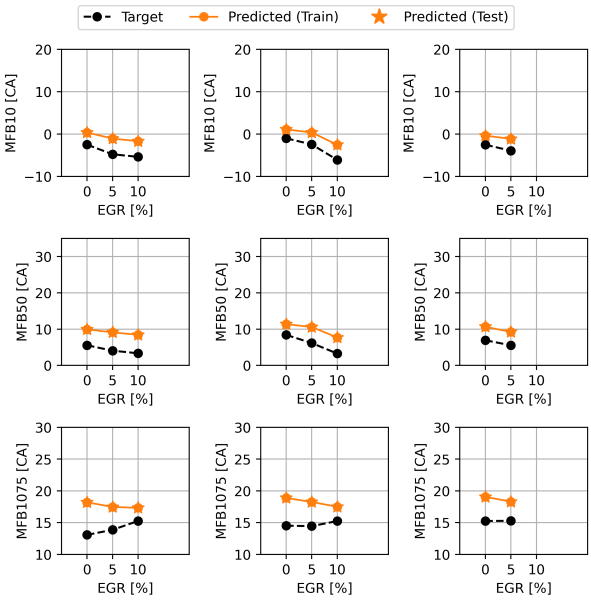



Figure 4.14: Combustion metrics prediction - sweep of $\lambda$ at 1500 RPM $\times$ 5.5 bar BMEP - Neural Network for burn rate prediction - Case 2

# Chapter 5

# Conclusions

This dissertation has introduced three machine learning models aimed to predict the combustion process of an ultra-lean dual-dilution spark-ignition engine. The models were developed using the experimental dataset obtained during the PHOENICE 2020 project.

The first model, a neural network for Wiebe parameter prediction, proved effective in estimating the parameters extracted using the genetic algorithm. However, the accuracy of the reconstructed combustion profile, when compared to the experimental one, was limited by the difficulties encountered by the GA in correctly fitting the original curve under the challenging conditions of ultra-lean combustion.

The second model, based on Gaussian Process Regression (GPR), demonstrated strong interpolation capabilities even with a reduced number of fitting points (less than 7% of the total). It achieved higher accuracy in predicting combustion metrics such as MFB10 and MFB50 compared to the Wiebe-based approach. Nevertheless, its performance declined when applied to unseen operating conditions, and its computational cost increases significantly with larger datasets.

Finally, the third model, a fully connected neural network for burn rate prediction, achieved the best balance between accuracy, robustness, and computational efficiency. With a relatively compact architecture, it was able to generalize well to

both training and test datasets, outperforming the other approaches particularly in the prediction of combustion duration. Although its accuracy decreased when tested on entirely unseen sweeps, the model retained the ability to capture the main combustion trends.

In summary, the analysis confirmed that machine learning–based approaches are well suited to model advanced combustion strategies in spark-ignition engines, where traditional 0D/1D methods struggle with calibration flexibility and computational cost. Among the tested models, the neural network for burn rate prediction proved to be the most promising solution, thanks to its high accuracy, relatively low computational time, and ability to generalize beyond the training data.

Future research should focus on expanding the dataset to cover a broader operating map and integrating the proposed models into real-time virtual calibration environments. These developments would further enhance the applicability of machine learning in supporting the design of next-generation, high-efficiency internal combustion engines.

# Bibliography

[1] J. Dornoff, "Co2 emission standards for new passenger cars and vans in the european union," International Council on Clean Transportation (ICCT), Policy Update, May 2023. [Online]. Available: https://theicct.org/publication/eu-co2-standards-cars-vans-may23/

[2] A. Suyabodha, A. Pennycott, and C. J. Brace, "A preliminary approach to simulating cyclic variability in a port fuel injection spark ignition engine," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 227, no. 5, pp. 665–674, 2013.

[3] A. Kakaee, M. Shojaeefard, and J. Zareei, "Sensitivity and effect of ignition timing on the performance of a spark ignition engine: an experimental and modeling study," *Journal of Combustion*, vol. 2011, no. 1, p. 678719, 2011.

[4] IBM. (2025) Overfitting vs. underfitting. IBM Think. Accessed: 2025-09-02. [Online]. Available: https://www.ibm.com/think/topics/overfitting-vs-underfitting

[5] MathWorks. (2025) What is overfitting? MathWorks. Accessed: 2025-09-02. [Online]. Available: https://it.mathworks.com/discovery/overfitting.html

[6] M. Wood. (2021) Simple model explains how brain cells connect. https://biologicalsciences.uchicago.edu/news/simple-model-brain-cells-connect. Accessed: 2025-09-01.

[7] P. Petru. (2025, Jan.) What is a neural network? a deep dive. Roboflow Blog. Accessed: 2025-09-02. [Online]. Available: https://blog.roboflow.com/what-is-a-neural-network/

[8] M. A. Albadr, S. Tiun, M. Ayob, and F. Al-Dhief, "Genetic algorithm based on natural selection theory for optimization problems," *Symmetry*, vol. 12, no. 11, p. 1758, 2020.

[9] C. Davies and R. Harms, "Report on the community strategy to reduce $co_2$ emissions from passenger cars and light-commercial vehicles," Committee on the Environment, Public Health and Food Safety, European Parliament, Tech. Rep. 2007/2119(INI), Sep 2007, rapporteur: Chris Davies. Draftsman: Rebecca Harms, Committee on Industry, Research and Energy.

[10] C. Sayin, H. M. Ertunc, M. Hosoz, I. Kilicaslan, and M. Canakci, "Performance and exhaust emissions of a gasoline engine using artificial neural network," *Applied thermal engineering*, vol. 27, no. 1, pp. 46–54, 2007.

[11] N. K. Togun and S. Baysec, "Prediction of torque and specific fuel consumption of a gasoline engine by using artificial neural networks," *Applied Energy*, vol. 87, no. 1, pp. 349–355, 2010.

[12] H. Yuan, H. Goyal, R. Islam, K. Giles, S. Howson, A. Lewis, D. Parsons, S. Esposito, S. Akehurst, P. Jones *et al.*, "Thermodynamics-based data-driven combustion modelling for modern spark-ignition engines," *Energy*, vol. 313, p. 134074, 2024.

[13] J. B. Heywood, "Combustion engine fundamentals," *1ª Edição. Estados Unidos*, vol. 25, pp. 1117–1128, 1988.

[14] A. Kalwar and A. K. Agarwal, "Overview, advancements and challenges in gasoline direct injection engine technology," in *Advanced combustion techniques and engine technologies for the automotive sector*. Springer, 2019, pp. 111–147.

[15] *GT-SUITE Engine Performance Manual*, Gamma Technologies, 2022.

[16] IBM, "What is artificial intelligence (ai)?" https://www.ibm.com/think/topics/artificial-intelligence, accessed: 2025-09-01.

[17] ——, "The history of ai," https://www.ibm.com/think/topics/history-of-artificial-intelligence, accessed: 2025-09-01.

[18] ——, "10 everyday machine learning use cases," https://www.ibm.com/think/topics/machine-learning-use-cases, accessed: 2025-09-01.

[19] ——, "Supervised versus unsupervised learning: What's the difference?" https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning, accessed: 2025-09-01.

[20] ——, "What is reinforcement learning?" https://www.ibm.com/think/topics/reinforcement-learning, accessed: 2025-09-01.

[21] ——, "What is feature selection?" https://www.ibm.com/think/topics/feature-selection, accessed: 2025-09-01.

[22] ——, "What is feature engineering?" https://www.ibm.com/think/topics/feature-engineering, accessed: 2025-09-01.

[23] GeeksforGeeks, "Gaussian process regression," https://www.geeksforgeeks.org/machine-learning/gaussian-process-regression-gpr/, accessed: 2025-09-01.

[24] scikit learn, "Gaussian process regression," https://scikit-learn.org/stable/modules/gaussian_process.html, accessed: 2025-09-01.

[25] IBM, "What is a neural network?" https://www.ibm.com/think/topics/neural-networks, accessed: 2025-09-01.

[26] ——, "Ai vs. machine learning vs. deep learning vs. neural networks: What's the difference?" https://www.ibm.com/think/topics/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks, accessed: 2025-09-01.

[27] Predict. (2021) Artificial neural networks: Mapping the human brain. https://medium.com/predict/artificial-neural-networks-mapping-the-human-brain-2e0bd4a93160. Accessed: 2025-09-01.

[28] K.-L. Du and M. N. Swamy, *Neural networks and statistical learning.* Springer Science & Business Media, 2013.

[29] GeeksforGeeks, "Layers in artificial neural networks," https://www.geeksforgeeks.org/deep-learning/layers-in-artificial-neural-networks-ann/, accessed: 2025-09-01.

[30] E. Sankumaravel. (2023) Learning algorithms in artificial neural networks. Medium. Accessed: 2025-09-02. [Online]. Available: https://medium.com/@elanesankumaravel/learning-algorithms-in-artificial-neural-networks-9ae39c9c78ec

[31] C. D. Marino, G. Maiorana, P. Pallotti, S. Quinto *et al.*, "The global small engine 3 and 4 cylinder turbo: The new fca's family of small high-tech gasoline engines," in *39th International Vienna Motor Symposium*, Vienna, Austria, Apr. 2018.

[32] L. Bernard, A. Ferrari, D. Micelli *et al.*, "Electro-hydraulic valve control with multiair technology," *MTZ Worldwide*, 2009.

[33] T. Tahtouh, M. André, G. Castellano, L. Rolando, and F. Millo, "A path toward a new generation of sustainable spark ignition engines: Experimental investigations on the synergic use of dual diluted combustion and renewable fuels," *Transportation Engineering*, vol. 20, p. 100317, 2025.

[34] Scribbr. Pearson correlation coefficient (r). Accessed: 2025-09-03. [Online]. Available: https://www.scribbr.com/statistics/pearson-correlation-coefficient/

[35] L. F. Scabini and O. M. Bruno, "Structure and performance of fully connected neural networks: Emerging complex network properties," *Physica A: Statistical Mechanics and its Applications*, vol. 615, p. 128585, 2023.