



UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE ENGENHARIA MECÂNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

Ruan Cardoso Comelli

**Automated model selection for pool boiling heat transfer estimation on
multiple surfaces**

Florianópolis
2023

Ruan Cardoso Comelli

**Automated model selection for pool boiling heat transfer estimation on
multiple surfaces**

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Engenharia Mecânica da Universidade Federal de Santa Catarina para a obtenção do título de Mestre em Engenharia Mecânica.

Orientador: Prof. Alexandre Kupka da Silva, PhD.

Florianópolis

2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Comelli, Ruan Cardoso

Automated model selection for pool boiling heat
transfer estimation on multiple surfaces / Ruan Cardoso
Comelli ; orientador, Alexandre Kupka da Silva, 2023.
133 p.

Dissertação (mestrado) - Universidade Federal de Santa
Catarina, Centro Tecnológico, Programa de Pós-Graduação em
Engenharia Mecânica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Mecânica. 2. Ebulição em piscina. 3.
Aprendizado de máquina automatizado. 4. Redes neurais
convolucionais. I. da Silva, Alexandre Kupka. II.
Universidade Federal de Santa Catarina. Programa de Pós
Graduação em Engenharia Mecânica. III. Título.

Ruan Cardoso Comelli

**Automated model selection for pool boiling heat transfer estimation on
multiple surfaces**

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca
examinadora composta pelos seguintes membros:

Prof. Danilo Silva, Ph.D.
Universidade Federal de Santa Catarina

Prof. Júlio César Passos, Dr.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi
julgado adequado para obtenção do título de Mestre em Engenharia Mecânica.

Prof. Henrique Simas, Dr.
Coordenador do Programa de
Pós-Graduação em Engenharia Mecânica

Prof. Alexandre Kupka da Silva, PhD.
Orientador

Florianópolis, 2023.

Dedico este trabalho aos meus queridos pais, que sempre me apoiaram e acreditaram em mim, mesmo quando eu não sabia aonde ir.

AGRADECIMENTOS

Este trabalho não teria sido chegado ao fim sem o esforço imensurável de inúmeras pessoas.

Em primeiro lugar, gostaria de agradecer ao meu orientador, o prof. Alexandre Kupka da Silva, por todo o apoio e a confiança ao longo desses anos. Sei que não foi fácil, e que muitas vezes parecia que tudo daria errado. Ainda assim, o prof. Kupka confiou em mim e me deu a liberdade de seguir o meu caminho. Por isso, sou grato.

Sou grato também ao Edevaldo, que me auxiliou na adaptação da bancada experimental e insistentemente buscou formas de medir a temperatura da superfície de ebulição. Não deu certo, mas aprendemos muito. Estendo este agradecimento ao Scussel, que, em uma inspeção de cinco minutos, detectou inúmeras falhas na bancada e ajudou a corrigi-las.

Também agradeço aos membros da banca examinadora, os profs. Danilo Silva e Júlio Passos, que dedicaram seu tempo para ler e avaliar esta dissertação. Além disso, os conhecimentos adquiridos em suas aulas foram fundamentais para o desenvolvimento deste trabalho.

Sou imensamente grato aos meus amigos e colegas do LEPTEN, que compartilharam momentos de alegria e de tristeza, e que sempre estiveram dispostos a ajudar e a debater ideias para melhorar o trabalho. Adriano, Arthur, Felipe, Gio, Luiz, Olivia, Paulo, Victor, e, em especial, Alex, Bruno e Thaís: obrigado por tudo!

Gostaria de agradecer também à CAPES, pela bolsa de mestrado, à UFSC, por toda a infraestrutura, e ao POSMEC, pela oportunidade.

Por fim, e acima de tudo, agradeço à minha família por todo o apoio e incentivo. Por todo o seu amor, e pelos sacrifícios que vocês fizeram para que eu pudesse me dedicar ao mestrado. Pai, mãe, Nati, Bia e vó: sem vocês, nada disso seria possível. Vô, sentirei sua falta para sempre.

*Yes, there are two paths you can go by, but in the long run
There's still time to change the road you're on*
Page and Plant (1971)

RESUMO

A ebulição em piscina nucleada é um dos processos de transferência de energia mais eficientes encontrados na indústria. Entretanto, a ebulição requer monitoramento e controle rigorosos para evitar a crise de ebulição. Técnicas de medição comumente empregadas são caras e requerem manutenção frequente, enquanto modelos numéricos têm incertezas elevadas associadas. Recentemente, algoritmos de aprendizado de máquina foram aplicados para estimar o fluxo de calor na ebulição em piscina a partir de imagens de baixa resolução. No entanto, a generalização desses modelos para diferentes superfícies de ebulição e a otimização de sua arquitetura ainda são questões em aberto. Este estudo explora esses aspectos, investigando o desempenho de modelos treinados e avaliados com diferentes superfícies aquecedoras. Uma nova metodologia baseada em aprendizado de máquina automatizado foi proposta para encontrar arquiteturas performáticas de forma eficiente. Uma bancada de ebulição em piscina foi adaptada para gerar dados de treinamento utilizando fios e fitas como superfícies aquecedoras. Medidas do fluxo de calor e imagens do processo de ebulição foram coletadas durante cada experimento. Quatro conjuntos de dados foram gerados empregando dois fios de diâmetros distintos e duas fitas em posições diferentes. As fases de pré-processamento de dados e de treinamento foram implementadas em Python com a biblioteca TensorFlow, e foram validadas treinando-se modelos com uma arquitetura de base e comparando-se o seu desempenho com estudos de referência. O aprendizado de máquina automatizado utilizou o algoritmo guloso disponível na biblioteca AutoKeras para buscar arquiteturas de redes neurais convolucionais e hiperparâmetros de treinamento que minimizassem o erro de predição sem aumentar o tamanho do modelo. O primeiro grupo de resultados consistiu na validação e na otimização da sequência de pré-processamento. Observou-se que a padronização das imagens melhora o desempenho dos modelos em até 47%. Posteriormente, o desempenho de modelos treinados e avaliados com diferentes superfícies foi investigado. Os resultados mostraram que os modelos foram incapazes de generalizar corretamente para superfícies não vistas no seu conjunto de treinamento. Por isso, o conjunto de treinamento deve conter exemplos de todas as superfícies de interesse para alcançar um desempenho aceitável. Modelos treinados e avaliados nas quatro superfícies de ebulição obtiveram resultado superior à maioria das correlações numéricas da literatura. Por fim, um último grupo de resultados empregou o aprendizado de máquina automatizado para encontrar arquiteturas mais performáticas e minimizar o erro de predição. Os modelos encontrados tiveram desempenho significativamente superior aos modelos de base, reduzindo o erro de validação em até 80%. Além disso, os modelos encontrados são até 30 vezes menores que os modelos de base, uma característica essencial para aplicações em tempo real. Quando treinados e avaliados sobre uma mesma superfície, os modelos tiveram performance superior à melhor correlação numérica encontrada na literatura.

Palavras-chave: Ebulição em piscina. Aprendizado de máquina automatizado. Redes neurais convolucionais.

RESUMO EXPANDIDO

Introdução

A crescente demanda por energia em escala global motiva a busca por métodos gradativamente mais eficientes de se gerar, transportar e armazenar energia, objetivando minimizar os seus impactos ambientais e econômicos. Nesse contexto, a ebulição demonstra ser um dos mecanismos de troca de calor mais eficientes devido ao elevado fluxo de calor que se obtém com pequenas diferenças de temperatura. Por isso, a ebulição é encontrada como o principal mecanismo de transporte de energia em diversas aplicações industriais. Em contrapartida, sistemas que operam com ebulição requerem intenso monitoramento e controle para garantir a máxima eficiência e segurança operacional. Entretanto, técnicas de medição bem estabelecidas são intrusivas, de elevado custo e necessitam de frequente manutenção, ao passo que correlações numéricas que modelam o processo de ebulição são limitadas em sua aplicabilidade e estão geralmente associadas a elevadas incertezas, de até $\pm 100\%$. Tendo isso em vista, estudos recentes aplicaram algoritmos de aprendizado de máquina para estimar o fluxo de calor dissipado na ebulição em piscina a partir de imagens de baixa resolução obtidas com uma câmera convencional. Demonstrou-se que redes neurais convolucionais são capazes de estimar o fluxo de calor com erros tão baixos quanto $\pm 7.37\%$. Apesar do resultado promissor, esses estudos se limitaram a estudar uma única superfície de ebulição e não exploraram a influência da arquitetura dos modelos no seu desempenho. Dessa forma, a aplicabilidade de tais modelos em outras condições de operação e o máximo desempenho que podem alcançar ainda são questões em aberto. O presente trabalho objetiva explorar tais aspectos. Especificamente, buscou-se investigar e otimizar o desempenho de redes neurais convolucionais quando treinadas e avaliadas em diferentes superfícies de ebulição, bem como estudar a influência da arquitetura dos modelos no seu desempenho. Para tanto, uma nova metodologia baseada em aprendizado de máquina automatizado foi proposta para a busca eficiente de arquiteturas de redes neurais convolucionais performáticas.

Metodologia

A fim de gerar os conjuntos de dados empregados no treinamento de redes neurais, uma bancada de ebulição em piscina foi adaptada para utilizar diferentes fios e fitas como superfícies de ebulição. Os experimentos consistiram na ebulição saturada de água deionizada à pressão atmosférica sobre diferentes superfícies de ebulição. Para forçar a ebulição, corrente elétrica foi aplicada sobre o sistema em níveis crescentes, partindo de zero, até que a seção de teste falhasse pela crise de ebulição. Dessa forma, cada conjunto de dados contém exemplos dos regimes de convecção natural, ebulição nucleada parcial, ebulição nucleada totalmente desenvolvida e ebulição em filme. Ao longo de cada experimento, medidas do fluxo de calor foram coletadas por um sistema de aquisição de dados, e pareadas com imagens do processo de ebulição adquiridas por uma câmera convencional. A partir desse procedimento, quatro conjuntos de dados foram gerados utilizando diferentes seções de teste: dois fios de diâmetros diferentes e duas fitas instaladas nas posições vertical e horizontal. O conjunto de dados obtido a partir do fio de maior diâmetro é o que mais se assemelha ao conjunto de dados utilizado em estudos anteriores, e por isso foi utilizado como base para a avaliação do desempenho dos modelos sobre as demais superfícies.

Empregando os conjuntos de dados gerados, redes neurais convolucionais foram treinadas para estimarem o fluxo de calor dissipado na ebulição em piscina, a partir das imagens de

baixa resolução obtidas nos experimentos. As etapas de pré-processamento de dados e de treinamento de modelos foram implementadas em Python 3.10, empregando a biblioteca de código aberto TensorFlow 2.10, e validadas com base em estudos anteriores. Na etapa de pré-processamento, as imagens de ebulição foram cortadas, transformadas em escala de cinza e reduzidas a fim de diminuir a dimensionalidade dos dados. Além disso, como motivado por um dos resultados, as imagens também foram padronizadas para que a média e o desvio padrão da luminância de seus pixels fossem iguais a zero e um, respectivamente. O método de treinamento padrão consistiu em treinar uma rede neural convolucional com a arquitetura de referência por 100 épocas utilizando o otimizador Adam e o erro quadrático médio como a função de perda. Por sua vez, o aprendizado de máquina automatizado utilizou o algoritmo guloso disponível na biblioteca AutoKeras 1.0.20 para buscar arquiteturas de redes neurais convolucionais e hiperparâmetros de treinamento que minimizassem o erro de predição sem aumentar o tamanho do modelo.

Resultados

Com base na metodologia proposta, três grupos de resultados foram obtidos. O primeiro grupo de resultados estudou as etapas de pré-processamento de imagens buscando validar as escolhas feitas e maximizar o desempenho dos modelos treinados. Em especial, mostrou-se que a padronização das imagens é capaz de melhorar o desempenho dos modelos em até 47%.

O segundo grupo de resultados investigou o desempenho de modelos treinados e avaliados em diferentes superfícies de ebulição. Observou-se que modelos treinados em visualização direta, quando a superfície de ebulição é visível, têm performance significativamente superior à de modelos treinados na visualização indireta, quando a superfície de ebulição é removida das imagens. Além disso, mostrou-se que os modelos são incapazes de generalizar corretamente para superfícies de ebulição ausentes do seu conjunto de treinamento. Assim, para alcançar desempenho aceitável, é necessário que o conjunto de treinamento contenha exemplos de todas as superfícies de ebulição de interesse. Com base nisso, modelos foram treinados utilizando imagens das quatro superfícies de ebulição, obtendo resultado superior ao da maioria das correlações numéricas encontradas na literatura.

Por fim, o terceiro grupo de resultados empregou aprendizado de máquina automatizado para buscar arquiteturas mais performáticas e minimizar o erro de predição. Os modelos obtidos tiveram desempenho significativamente superior ao dos modelos de base. Na visualização direta, o erro de validação foi reduzido em 73%, ao passo que, na visualização indireta, o erro foi reduzido em 80%. Quando treinados e avaliados sobre uma única superfície, os modelos encontrados tiveram performance superior à da melhor correlação numérica encontrada na literatura, representando uma conquista significativa deste trabalho. Além disso, os modelos obtidos são consideravelmente menores que os modelos de base, o que pode ser útil para aplicações em tempo real.

Palavras-chave: Ebulição em piscina. Aprendizado de máquina automatizado. Redes neurais convolucionais.

ABSTRACT

Nucleate pool boiling is one of the most efficient energy transfer processes found in the industry. However, it requires close monitoring and control to avoid a boiling crisis. Commonly employed measurement techniques are expensive and require frequent maintenance, whereas numerical models have high associated uncertainties. Machine learning algorithms have recently been applied to estimate the heat flux in boiling pools from low-resolution images. However, the generalization of these models for different boiling surfaces and the optimization of their architecture are still open questions. This study explores these aspects, investigating the performance of models trained and evaluated with different heating surfaces. A new methodology based on automated machine learning was proposed to find performing architectures efficiently. A pool boiling bench was adapted to generate training data using wires and ribbons as heating surfaces. Heat flow measurements and images of the boiling process were collected during each experiment. Four datasets were generated using two wires of different diameters and two ribbons in different positions. The data preprocessing and training pipelines were implemented in Python with the TensorFlow library. They were validated by training models with a baseline architecture and comparing their performance with reference studies. The automated machine learning pipeline used the greedy algorithm from the AutoKeras library to search for convolutional neural network architectures and training hyperparameters that minimized the prediction error without increasing the model size. The first group of results consisted of validating and optimizing the preprocessing pipeline. It was observed that image standardization improves the models' performance by up to 47%. Subsequently, the performance of models trained and evaluated with different heater surfaces was investigated. The results showed that the models could not generalize correctly to surfaces not seen in their training set. Therefore, the training set must contain examples of all surfaces of interest to achieve acceptable performance. Models trained and evaluated on all four boiling surfaces obtained better results than most numerical correlations in the literature. Finally, a last group of results employed automated machine learning to find more performant architectures and minimize prediction error. The models performed significantly better than the baseline models, reducing the validation error by up to 80%. Furthermore, the models found by automated machine learning are up to 30 times smaller than the baseline models, an essential characteristic for real-time applications. When trained and evaluated on the same surface, the models performed better than the best numerical correlation found in the literature.

Keywords: Pool boiling. Automated machine learning. Convolutional neural networks.

LIST OF FIGURES

Figure 1 – Pool boiling photographs obtained with a nichrome wire with a diameter of 0.5 mm and length of 6.5 cm at different heat flux levels.	36
Figure 2 – Boiling curve obtained by controlling the heat flux. Four regimes are highlighted: natural convection, partial nucleate boiling, fully developed nucleate boiling, and film boiling. For the three first regimes, an example frame is presented. Five characteristic points are also shown: the origin (O), the ONB, the transition point (B), the DNB, and the burnout point. The x-y data points utilized to plot the boiling curve were extracted from Çengel and Ghajar (2009).	37
Figure 3 – Visible parametric effects. Each row illustrates how an isolated effect changes the observable behavior of the system.	40
Figure 4 – Boiling curves for small, intermediate, and large wire diameters. To the left (a), illustrations represent wires with small and large diameters. To the right (b), the observed boiling curves obtained from those wires and an intermediate sample are shown. The x-y data points utilized to plot the boiling curves (b) were extracted from J. H. Kim, You, and Pak (2006).	41
Figure 5 – Boiling curves for horizontal and vertical heater surfaces. To the left (a), illustrations represent a flat heater surface in the horizontal and vertical positions. To the right (b), the observed boiling curves obtained from those positions are presented. The x-y data points utilized to plot the boiling curves (b) were extracted from Rohsenow, Hartnett, and Cho (1998).	42
Figure 6 – Illustrations of (a) an MLP and (b) a unit.	49
Figure 7 – Demonstration of the application of a convolutional layer filter. Each element in the output matrix $\mathbf{I} * \mathbf{K}$ is calculated as the dot-product between the corresponding window in the input matrix \mathbf{I} and the kernel \mathbf{K} . The values in the matrices were extracted from Neutelings (2022).	51
Figure 8 – Demonstration of the application of a 2×2 max-pooling operation. Each output pixel is the maximum of the corresponding 2×2 window in the input image.	53
Figure 9 – Illustration of a convolutional neural network containing two convolutional layers, each followed by a pooling layer. The convolutional layers are followed by a hidden, dense layer and topped by the output layer.	53

Figure 10 – Representation of the CNN utilized by Hobold and da Silva (2019b). A convolutional block is a sequence of three layers: a convolutional layer, an activation layer, and a max-pooling layer. Similarly, a dense block is a dense layer followed by an activation layer. Each layer is displayed alongside its outputs' shapes and the number of trainable weights it contributes to the model. The weights of dense layers are arranged in one-dimensional vectors whose shape is denoted as $(\cdot,)$, following the Python notation for tuples of unitary length.	59
Figure 11 – Representation of the CNN utilized by Scariot (2019). The same notes for Figure 10 apply.	60
Figure 12 – The areas of study where this work intersects.	62
Figure 13 – Pool boiling setup, including (a) a photograph demonstrating the positioning and scale of the components of the experimental apparatus; (b) a schematic representation of the boiling chamber showing the test sample connected to the electrodes, the path of the electric current I and the auxiliary subsystems; and (c) an illustration of the visualization equipment displaying the relative positioning of the camera, the boiling chamber, the light diffuser, and the backlighting lamp. Note that (b) and (c) are not to scale.	64
Figure 14 – Electrical diagram for the pool boiling experimental apparatus.	65
Figure 15 – Illustration of the experimental procedure, including (a) multiple thermal power steps increasing in steps of 5 W/cm^2 until burnout, and (b) the sequence of thermal power stabilization, LED synchronization, and video recording. Note that this illustration is conceptual and does not contain real experimental data.	68
Figure 16 – Example video frame obtained at 75 W for the baseline, large wire dataset \mathcal{D}^{LW} . The video frame contains environmental clues such as the LED or reflexes, which are cropped out of the ROI, as described in Section 4.2.2.	70
Figure 17 – Demonstration of the results of the preprocessing pipeline, with (a) the original image and (b) its preprocessed version.	78
Figure 18 – Demonstration of the preprocessing step of cropping an image to the ROI. Obtained from the large wire pool boiling dataset at the power level of 30 W.	79
Figure 19 – Demonstration of the preprocessing step of grayscaling an image in matrix representation. The input image is encoded in 8-bit unsigned integer type and its color channels are shown as separate matrices for easier understanding.	81

Figure 20 – Demonstration of the preprocessing step of grayscaling an image. Obtained from the large wire pool boiling dataset at the power level of 30 W. There is little perceivable difference between (a) the colored and (b) the grayscale images.	81
Figure 21 – Demonstration of the preprocessing step of downscaling an image with a downscaling factor of $f_{ds} = 2$, resulting in a $f_{ds}^2 = 4$ dimensionality reduction. Each pixel in the output image is the average of the elements in the corresponding 2×2 window in the input image.	82
Figure 22 – Downscaling analyses of (a) the relative variance and (b) the cross-entropy ratio as functions of the downscaling factor f_{ds} for the four pool boiling datasets.	83
Figure 23 – Demonstration of the preprocessing step of downscaling an image using a downscaling factor of $f_{ds} = 5$. Obtained from the large wire pool boiling dataset at the power level of 30 W. There is little perceivable difference between (a) the full-scale and (b) the downscaled images, but the dataset dimensionality is reduced in $f_{ds}^2 = 25$ times.	84
Figure 24 – Demonstration of the preprocessing step of size uniformization and visualization cropping, either (b) keeping the heater surface in the pool boiling images in direct visualization; or (c) removing it in indirect visualization.	86
Figure 25 – Demonstration of the preprocessing step of horizontal visualization window cropping for (a) direct and (b) indirect visualization.	87
Figure 26 – Demonstration of the preprocessing step of standardizing images. In this example, a 3×3 grayscale image is standardized so that its mean and variance equal 0 and 1, respectively.	88
Figure 27 – Structural similarity between consecutive frames for the four pool boiling datasets. Generated by calculating the structural similarity index between a reference frame, identified as frame 0, and the subsequent 100 frames.	88
Figure 28 – Image brightness distribution in the baseline, large wire dataset in direct visualization. The result is presented as a letter-value plot (HOFMANN; KAFADAR; WICKHAM, 2011; WASKOM, 2021) with the removal of outliers. The distribution is shown for each nominal power level q for the training, validation and test subsets.	97
Figure 29 – Image downscaling results presented as the loss (MSE) seen as a function of the downscaling factor for the training, validation, and test subsets of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$	98

Figure 30 – Visualization window size results presented as the loss seen as a function of the visualization window fraction for the training, validation, and test subsets of the baseline, large wire dataset for $q'' \geq 10 \text{ W/cm}^2$. The results are shown for both (a) direct and (b) indirect visualization modes. 100

Figure 31 – Learning curve results presented as the loss seen as a function of the training dataset subsampling fraction for the training, validation, and test subsets of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$. Training metrics are evaluated on the subsampled dataset, whereas validation and test metrics include 100 % of the data. The results are shown for both (a) direct and (b) indirect visualization modes. 101

Figure 32 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in direct visualization for $q'' \geq 10 \text{ W/cm}^2$. The vertical axis represents the training set; thus, all results in the same row refer to the same model, trained on the training subset of the dataset denoted at the right of that row. Each column denotes a different evaluation set, denoted at the top of the plot. Each cell shows the validation loss (i.e., the MSE) at its top and the MAPE at its bottom. 106

Figure 33 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$. The same notes for Figure 32 are applicable. 107

Figure 34 – Diagram of the best model found by the AutoML pipeline in direct visualization. The convolutional blocks are sequences of three layers: a convolutional layer, an activation layer and a max-pooling layer. Similarly, the dense blocks are dense layers followed by activation layers. The input layer is annotated as standardized since image standardization was applied. Each layer is displayed alongside the shape of its outputs and the number of trainable weights it contributes to the model. Note that the weights of dense blocks are arranged in one-dimensional vectors whose shape is denoted as (\cdot) , following the Python notation for tuples of unitary length. 112

Figure 35 – Diagram of the best model found by the AutoML pipeline in indirect visualization. The same observations for Figure 34 are applicable. . . . 113

Figure 36 – Validation loss versus model size for both visualization modes. 113

Figure 37 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in direct visualization for $q'' \geq 10 \text{ W/cm}^2$. Models were instantiated from the best architecture found by the AutoML pipeline. The vertical axis represents the training set; thus, all results in the same row refer to the same model, trained on the training subset of the dataset denoted at the right of that row. Each column denotes a different evaluation set, denoted at the top of the plot. Each cell shows the validation loss (i.e., the MSE) at its top and the MAPE at its bottom.	115
Figure 38 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$. The same notes for Figure 37 are applicable.	116
Figure 39 – Data sheet for the test wires NI80-020-200 (Omega Engineering) and NI80-010-200 (Omega Engineering). Reproduced from Omega Engineering (2022b,c).	129
Figure 40 – Data sheet for the test ribbon NCR-17-100 (Omega Engineering) (part 1). Reproduced from Omega Engineering (2022a).	130
Figure 41 – Data sheet for the test ribbon NCR-17-100 (Omega Engineering) (part 2). Reproduced from Omega Engineering (2022a).	131

LIST OF TABLES

Table 1 – Reference metrics from Hobold and da Silva (2019b). Obtained for nucleate and film boiling regimes defined by $q'' \geq 10 \text{ W/cm}^2$. Validation and test metrics are presented as functions of the type of visualization: direct, in which the heater surface is visible in the images, or indirect, in which it is cropped out. Confidence intervals of 95% around the mean were computed by bootstrapping using 1000 batches of 100 samples. Reproduced from Hobold and da Silva (2019b).	58
Table 2 – Reference metrics from Scariot (2019). Obtained for two different datasets generated by choosing different heat flux levels from the global dataset. Validation and test metrics are computed by bootstrapping 10 000 batches of 1000 samples each. Reproduced from Scariot (2019).	61
Table 3 – Representative frames from each dataset in the natural convection, partial nucleate boiling, and fully developed nucleate boiling regimes. Note that not all frames have the same size since the ROI changed between experimental runs.	71
Table 4 – Approximate ranges for the thermal power and heat flux at the ONB and the DNB for each experimental dataset. The main ranges are the nominal thermal power levels surrounding the event, whereas the values in parentheses are the average heat flux calculated for those levels. Because of natural variability in the measured heat flux, the averages in parentheses do not necessarily satisfy Equation (2). In addition, the higher ends of the DNB intervals are unknown since no experimental data is available for those points.	72
Table 5 – Maximum heat flux uncertainties for each experimental dataset. The standard and expanded uncertainty, obtained with a coverage factor of 2, are presented. For convenience, the simplified uncertainty, as it would appear in a measurement report, is also shown.	73
Table 6 – Dataset split sizes. Each subset (training, validation and test) is a disjoint subsample of the global dataset. The split sizes are shown along with the total dataset size for each experimental dataset.	77
Table 7 – Image shapes before and after the ROI cropping preprocessing step for each pool boiling dataset. Shapes are presented in the format height \times width \times color channels.	80
Table 8 – Image shapes before and after the grayscaling preprocessing step for each pool boiling dataset. Shapes are presented in the format height \times width \times color channels.	82

Table 9	– Image shapes before and after downscaling each pool boiling dataset by a factor of $f_{ds} = 5$. Shapes are presented in the format height \times width \times color channels.	84
Table 10	– Image shapes before and after height uniformization and vertical crop in direct and indirect visualization. Shapes are presented in the format height \times width \times color channels.	86
Table 11	– Training pipeline validation results. Metrics are presented only for nucleate and film boiling regimes, defined by $q'' \geq 10 \text{ W/cm}^2$. Metrics for the training, validation and test sets are presented as functions of the type of visualization: direct, in which the heater surface is visible in the images, or indirect, in which it is omitted. Confidence intervals of 95% around the mean were computed by bootstrapping 1000 batches of 1000 samples. The reference results are reproduced from Hobold and da Silva (2019b).	91
Table 12	– Automated machine learning search space. Each category of hyperparameters is displayed separately. The search space of each hyperparameter is shown as a discrete set of the possible values it might assume, or the set {NO, YES} for configurations that might be absent or present, respectively.	94
Table 13	– Image standardization performance metrics. Validation and test performance metrics of the baseline architecture trained with two versions of the default training pipeline: with and without the image standardization step. The metrics are presented for the direct and indirect visualization cases of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$	96
Table 14	– Training, validation, and test metrics for models trained and evaluated on each heater surface separately in direct visualization for $q'' \geq 10 \text{ W/cm}^2$. 102	
Table 15	– Training, validation, and test metrics for models trained and evaluated on each heater surface separately in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$. 103	
Table 16	– Merged datasets split sizes. Each subset (training, validation and test) is a disjoint subsample of the total dataset. The split sizes are shown along with the total dataset size of each dataset. The sizes for \mathcal{D}^{LW} , \mathcal{D}^{SW} , \mathcal{D}^{HR} and \mathcal{D}^{VR} are the same as in Table 6. The size of each merged dataset is the average of the sizes of its component datasets.	105
Table 17	– Best hyperparameters found by the AutoML pipeline in both visualization modes.	109
Table 18	– AutoML search results. Metrics are presented only for nucleate and film boiling regimes defined by $q'' \geq 10 \text{ W/cm}^2$. Metrics for the training, validation, and test sets are presented as functions of the visualization mode: direct or indirect.	110

LIST OF ABBREVIATIONS AND ACRONYMS

AutoML	Automated Machine Learning
CHF	Critical Heat Flux
CNN	Convolutional Neural Network
DAQ	Data Acquisition System
DC	Direct Current
DNB	Departure of Nucleate Boiling
DSLR	Digital Single-Lens Reflex
FoV	Field of View
GPU	Graphics Processing Unit
LED	Light-Emitting Diode
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
ONB	Onset of Nucleate Boiling
PCA	Principal Component Analysis
PoV	Point of View
PTFE	Polytetrafluoroethylene
RMSE	Root Mean Squared Error
ROI	Region of Interest
RTD	Resistance Thermometer
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine

LIST OF SYMBOLS

\mathcal{D}^{LW}	Baseline pool boiling dataset obtained from the large nichrome wire
\mathcal{D}^{SW}	Pool boiling dataset obtained from the small nichrome wire
\mathcal{D}^{HR}	Pool boiling dataset obtained from the horizontal ribbon
\mathcal{D}^{VR}	Pool boiling dataset obtained from the vertical ribbon
q''	Heat flux [W/cm^2]
h	Heat transfer coefficient [$\text{W}/(\text{cm}^2 \text{ } ^\circ\text{C})$]
T_∞	Bulk liquid temperature [$^\circ\text{C}$]
T_s	Heater surface temperature [$^\circ\text{C}$]
T_{sat}	Saturation temperature [$^\circ\text{C}$]
ΔT_{sat}	Wall superheat [$^\circ\text{C}$]
q	Heat transfer rate, or thermal power [W]
A_s	Heater surface area [cm^2]
μ	Dynamic viscosity [$\text{N s}/\text{m}^2$]
i_{lv}	Enthalpy of vaporization [J/kg]
g	Gravity [m/s^2]
ρ	Massic density [kg/m^3]
σ	Surface tension [N/m]
c_p	Specific heat at constant pressure [$\text{J}/(\text{kg } ^\circ\text{C})$]
Pr	Prandtl number
p^*	Reduced pressure
R	Specific gas constant [$\text{J}/(\text{kg K})$]
λ	Thermal conductivity [$\text{W}/(\text{m } ^\circ\text{C})$]
ν	Kinematic viscosity [m^2/s]
p	Pressure [Pa]
p_c	Critical pressure of a fluid [Pa]
N_a	Density of active nucleation sites [$1/\text{cm}^2$]
D_d	Bubble departure diameter [mm]
f	Frequency of bubble departure [1/s]
α	Thermal diffusivity [m^2/s]
t_w	Bubble waiting time [s]
θ	Contact angle
\mathcal{D}	Dataset
\mathbf{x}	Feature vector
y	Target
\mathbb{R}	The set of real numbers
$\mathcal{D}_{\text{train}}$	Training set
$\mathcal{D}_{\text{test}}$	Test set
\mathcal{D}_{val}	Validation set

ℓ_2	ℓ_2 penalization
$\hat{\mathbf{y}}$	Prediction vector
\mathbf{a}	Activation vector
ReLU	Rectified linear unit activation function
J	Cost function
L	Loss function
\mathbf{y}	Target vector
\hat{y}	Prediction
\mathbf{K}	Convolutional kernel
u	Activation function
R^2	R^2 score
\bar{y}	Average value of the targets
f_{ds}	Downscaling factor
V_{sample}	Voltage drop at the test sample [V]
I	Electric current [A]
V_{shunt}	Voltage drop at the shunt [V]
R_{shunt}	Shunt resistance [Ω]
R_{sample}	Electric resistance of the test sample [Ω]
\hat{U}	Standard uncertainty
HIST	The 100-bin luminosity distribution of a frame
S	Cross-entropy between two histograms
$D^{f_{\text{ds}}}$	The downscaling operator associated with the downscaling factor f_{ds}
S^*	Relative cross-entropy between two histograms
VAR *	Relative variance between two histograms
VAR	Variance of a histogram
$\bar{\cup}$	Dataset merge operator

CONTENTS

1	INTRODUCTION	31
1.1	OBJECTIVES	32
1.2	DOCUMENT OUTLINE	33
2	LITERATURE REVIEW	35
2.1	POOL BOILING	35
2.1.1	Boiling curve	36
2.1.2	Parametric effects	39
2.1.3	Numerical correlations	43
2.1.4	Mechanistic models	45
2.2	MACHINE LEARNING	47
2.2.1	Deep learning	48
2.2.2	Convolutional Neural Networks	51
2.2.3	Performance metrics	54
2.2.4	Automated machine learning	55
2.3	MACHINE LEARNING-BASED ANALYSIS OF PHASE-CHANGE	56
2.4	PROBLEM SPECIFICATION	61
3	EXPERIMENTAL METHODOLOGY	63
3.1	EXPERIMENTAL APPARATUS	63
3.2	ADAPTATION OF THE EXPERIMENTAL APPARATUS	65
3.3	EXPERIMENTAL PROCEDURE	67
3.4	EXPERIMENTAL CASES	68
3.5	UNCERTAINTY ANALYSIS	73
3.6	TEMPERATURE MEASUREMENT	73
4	MACHINE LEARNING METHODOLOGY	75
4.1	IMPLEMENTATION AND EXECUTION	76
4.2	DATASETS	76
4.2.1	Dataset splitting	76
4.2.2	Image preprocessing	78
4.2.2.1	Image pixel format conversion	79
4.2.2.2	Region of interest cropping	79
4.2.2.3	Grayscale	81
4.2.2.4	Downscaling	82
4.2.2.5	Size uniformization	84
4.2.2.6	Vertical visualization cropping	85
4.2.2.7	Horizontal visualization window cropping	86
4.2.2.8	Standardization	87
4.2.3	Dataset subsampling	88

4.3	TRAINING	89
4.4	VALIDATION	90
4.5	AUTOMATED MACHINE LEARNING	92
5	RESULTS AND DISCUSSION	95
5.1	DATASET PREPROCESSING	95
5.1.1	Image standardization	95
5.1.2	Image downscaling	97
5.1.3	Visualization window size	99
5.1.4	Learning curve	100
5.2	MULTIPLE SURFACES	101
5.2.1	Single-surface evaluation	101
5.2.2	Multi-surface evaluation	104
5.3	AUTOMATED MACHINE LEARNING	108
5.3.1	Architecture search	108
5.3.2	Multi-surface evaluation	114
6	CONCLUSION	117
6.1	SUMMARY	117
6.2	RECOMMENDATIONS AND SUGGESTIONS FOR FUTURE STUDIES	118
	REFERENCES	121
	APPENDIX A – DATA SHEETS	129
A.1	NI80-020-200 AND NI80-010-200 (OMEGA ENGINEERING)	129
A.2	NCRR-17-100 (OMEGA ENGINEERING)	130
	APPENDIX B – UNCERTAINTY ANALYSIS	133

1 INTRODUCTION

Nucleate pool boiling is one of the most efficient heat transfer processes due to the high heat transfer rate obtained with a relatively small temperature difference, especially when compared to monophasic convection. Because of its high efficiency, boiling is fundamental in various applications: evaporators in refrigerators, heat exchangers, and steam generators (BERGMAN et al., 2011). Additionally, boiling plays a crucial role in nuclear power plant reactors since it is the primary heat transfer mechanism under normal operating conditions and can be also dominant in catastrophic scenarios (IAEA, 1999).

However, the design of boiling-driven thermal systems must often observe limitations imposed by the heater surface and operating conditions. For instance, most boiling systems have the Critical Heat Flux (CHF) as an upper bound for the heat transfer rate. When the heat flux is the controlled variable, heat fluxes higher than the CHF cause an abrupt increase in the surface temperature, potentially exceeding its operating limits and leading the system to fail in the *boiling crisis*. Therefore, monitoring the heat flux q'' is essential to ensure the safe operation of thermal systems. Moreover, quantifying the heat flux and the heat transfer coefficient h is indispensable to ensure the efficient operation of thermal systems (ÇENGEL; GHAJAR, 2009; BERGMAN et al., 2011).

Despite its critical role in thermal systems, boiling is a complex phenomenon whose modeling and simulation are difficult or, in many cases, almost impossible. Even well-established numerical correlations for estimating the heat transfer coefficient in pool boiling setups are associated with errors as high as $\pm 100\%$. Similarly, the CHF can be calculated with an uncertainty of up to 25% , as discussed in detail in Section 2.1.3. Given these limitations, most thermal systems require precise measurement subsystems to ensure their safe and efficient operation.

Recently, the application of Convolutional Neural Networks (CNNs) to the study of heat transfer processes has shown promising results. CNNs have been effectively employed to classify boiling regimes and quantify the heat transfer rate on low-resolution images of pool boiling from a nichrome wire (HOBOLD; DA SILVA, 2018b, 2019b). Compared to other well-established machine learning methods, such as Support Vector Machines (SVMs) and Multi-Layer Perceptrons (MLPs), CNNs demonstrated a significantly better performance in classifying pool boiling regimes. Furthermore, CNNs estimate heat flux with lower uncertainties than traditional techniques, such as most numerical correlations that take the temperature as input (HOBOLD; DA SILVA, 2019b).

Employing machine learning models to estimate heat fluxes in pool boiling setups is a promising approach that overcomes many limitations of traditional measurement systems. In particular, visualization-based models are non-intrusive, low-cost technologies that can be easily implemented in industrial applications. In addition, since machine learning models learn directly from data, they do not require domain expertise to be developed,

making them easier to extend to new scenarios and operating conditions. Despite those benefits, recent studies have not fully explored the impact of varying operating conditions on pool boiling behavior, limiting their overall applicability. A crucial open question in this field is the models' specificity to the boiling heater surface they are trained on and their ability to generalize to new, unseen surfaces.

To better understand that limitation, Scariot (2019) adapted the experimental apparatus from Hobold and da Silva (2018b,a, 2019a,b) to utilize bi-dimensional heating surfaces, obtaining models capable of estimating heat fluxes with less than 33% of relative error. While this is a promising result, it still falls short compared to the performance achieved by Hobold and da Silva (2019b), demonstrating that the performance of CNNs strongly depends on their architecture and training dataset.

The studies by Hobold and da Silva (2018b,a, 2019b,a) and Scariot (2019) highlight CNNs as a powerful, cost-effective, non-intrusive method for estimating heat flux in pool boiling systems from low-speed, low-resolution visualization. However, the unexplored aspects of those studies leave room for improvement in the performance of models. Notably, each study considered only a single heater surface and operating condition, limiting the generalization ability of the models. Moreover, Hobold and da Silva (2018b,a, 2019b,a) tested only three CNN architectures, whereas Scariot (2019) tested a single one, leaving room for further exploration and improvement in the architecture design.

The present work seeks to address the limitations of previous studies in applying CNNs for quantifying heat flux in pool boiling setups. The main focus of this work is to explore the influence of the heater surface on the performance of CNNs and to perform a systematic architectural search for optimal CNN architectures. Section 1.1 outlines the primary objective of this study and describes the associated specific objectives to be accomplished. With the potential demonstrated by CNNs in the classification and quantification of heat transfer in pool boiling, this work aims to further advance the understanding and utilization of CNNs in this area.

1.1 OBJECTIVES

The objective of this study is to assess the effectiveness of CNNs in modeling pool boiling processes in varying operating conditions. More precisely, the focus is on evaluating and optimizing the performance of CNNs in quantifying heat flux in pool boiling setups using image data collected from different heater surfaces.

This work aims to accomplish the following tasks to achieve the primary objective:

- Conduct a comprehensive review of existing literature on the use of visualization-based machine learning algorithms for monitoring pool boiling systems;
- Design and execute pool boiling experiments to gather image and heat flux data from different heater surfaces to train machine learning models;

- Develop data preprocessing and machine learning training pipelines for predicting heat flux from the pool boiling data;
- Validate the training pipeline by building and training a CNN in conditions similar to those reported in Hobold and da Silva (2019b) and verifying that equivalent performance is achieved;
- Optimize the preprocessing pipeline to improve the performance of CNNs;
- Investigate the performance of the CNNs in quantifying heat transfer in a pool boiling setup with various heater surfaces;
- Execute a systematic CNN architecture search to minimize prediction error.

1.2 DOCUMENT OUTLINE

This Thesis is divided into the following chapters to address the research objectives comprehensively:

Chapter 1 – Introduction: this introductory chapter provides the background and motivation for this research and outlines the objectives;

Chapter 2 – Literature review: presents a comprehensive literature review covering three main aspects: (a) the pool boiling phenomenon, including its fundamental aspects such as bubble nucleation and growth, the heat transfer mechanisms, and parametric effects, as well as the state-of-the-art numerical correlations and mechanistic models along with their associated uncertainties for later comparison with the machine learning models developed in this work; (b) machine learning fundamentals, encompassing deep learning, CNNs and Automated Machine Learning, with focus on the architecture search technique applied in this study; (c) application of machine learning algorithms to visually estimate heat flux in pool boiling setups, including previous studies such as Hobold and da Silva (2018b,a, 2019b,a) and Scariot (2019);

Chapter 3 – Experimental methodology: outlines the pool boiling experimental setup built as part of this work and details the experimental procedure used to gather the datasets utilized in this research;

Chapter 4 – Machine learning methodology: provides a comprehensive description of the data preprocessing and training pipelines employed in this research. The Chapter explains the design decisions behind each step, justifying them to ensure the reproducibility of results. Additionally, it presents the validation results demonstrating the correctness of the training pipeline. Finally, this Chapter explains the architecture search pipeline employed, which represents one of the main contributions of this work;

Chapter 5 – Results and discussion: presents and analyzes the results obtained in this work, covering the optimization of the data preprocessing pipeline, the evaluation of CNNs in quantifying heat flux in pool boiling setups with different heater surfaces, and the results of the automated architecture search. A thorough discussion of the findings is provided to highlight the key insights and implications of the study.

Chapter 6 – Conclusion: summarizes this research’s key findings and main conclusions. It includes a brief overview of this study’s main contributions and highlights the limitations and areas for improvement. This Chapter concludes by offering suggestions for future research that can build upon the work presented in this Thesis.

2 LITERATURE REVIEW

This Chapter provides a comprehensive literature review to lay the foundation for the research presented in this Thesis. The review includes fundamental concepts and recent publications closer to the state of the art. Section 2.1 focuses on pool boiling, the phase-change phenomenon of interest to this work, while Section 2.2 explains machine learning, the proposed solution for non-intrusive heat transfer measurement. Subsequently, Section 2.3 presents recent studies on the application of machine learning models to measure phase-change phenomena, emphasizing non-intrusive, visualization-based methods. Finally, Section 2.4 describes the aspects left unexplored by the literature and outlines the contributions this Thesis expects to bring to the understanding of the subject.

2.1 POOL BOILING

Boiling is a type of convective process associated with phase change. This process occurs when a liquid at temperature T_∞ enters into contact with a solid surface at a temperature T_s that exceeds the liquid saturation temperature, T_{sat} . If the wall superheat, $\Delta T_{\text{sat}} = T_s - T_{\text{sat}}$, is sufficiently high, bubble nucleation occurs (BERGMAN et al., 2011). In these circumstances, the heater surface transfers heat to the liquid at a *heat transfer rate* q . The *heat transfer coefficient*, defined as

$$h = \frac{q''}{\Delta T_{\text{sat}}}, \quad (1)$$

is a measure of the heat transfer process. In this equation,

$$q'' = \frac{q}{A_s} \quad (2)$$

is the *heat flux* transferred from the heater with surface area A_s to the liquid medium.

Pool boiling is, by definition, a mode of boiling in which a heater surface transfers heat to a quiescent liquid. Pool boiling happens under natural convection conditions. The transport of mass and energy is greatly intensified by the movement of vapor bubbles in the liquid after their nucleation, growth, and detachment from the heater surface. This mode of boiling differs from forced convection boiling (also known as flow boiling), which includes a forced, directed fluid motion. Pool boiling can be classified as either *saturated*, when the liquid is at its saturation temperature, $T_\infty = T_{\text{sat}}$, or *subcooled*, when $T_\infty < T_{\text{sat}}$ (BEJAN; KRAUS, 2003; BERGMAN et al., 2011; ÇENGEL; GHAJAR, 2009; KANDLIKAR, 1999). Pool boiling can be a very efficient form of heat transfer and is the primary mode of boiling in various energy conversion and heat exchange systems (KANDLIKAR, 1999). Bergman et al. (2011) point out that heat transfer coefficients higher than $10\,000\text{ W}/(\text{m}^2\text{ K})$ are characteristic of the nucleate pool boiling regime described in Section 2.1.1, which is considerably larger than most heat transfer processes with no phase change.

Figure 1 presents examples of pool boiling photographs at different dissipated heat flux levels. As it can be observed, the system's behavior changes visibly as the heat flux increases.

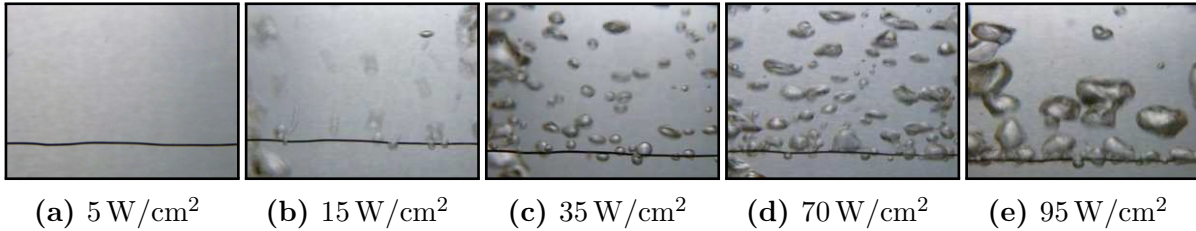


Figure 1 – Pool boiling photographs obtained with a nichrome wire with a diameter of 0.5 mm and length of 6.5 cm at different heat flux levels.

The following Sections describe the boiling curve, the effects that operating conditions have on the boiling process, and the most popular and widely used numerical correlations and mechanistic models for estimating heat transfer in pool boiling.

2.1.1 Boiling curve

Nukiyama (1966) was the first to identify the *boiling curve* using a pool boiling setup and a nichrome wire as the heater surface. In that study, the heat flux dissipated to the liquid, q'' , was controlled, and the associated wire surface superheat, ΔT_{sat} , was measured. The corresponding pairs $(\Delta T_{\text{sat}}, q'')$ constitute the boiling curve, which can be illustrated as in Figure 2 (BERGMAN et al., 2011; ÇENGEL; GHAJAR, 2009). Note that the boiling curve shown and described in this Section is obtained when the heat flux is the controlled variable and the temperature is measured. This is consistent with the experimental setup utilized in this work and many end industrial applications. For information about the alternative boiling curve obtained when the temperature is controlled and the heat flux is measured, consult Bergman et al. (2011), Çengel and Ghajar (2009), Rohsenow, Hartnett, and Cho (1998) and Kandlikar (1999).

The boiling curve obtained with the control of heat flux can be divided into segments, or regimes, with specific characteristics determined by the dynamics of vapor bubble formation (BERGMAN et al., 2011; ÇENGEL; GHAJAR, 2009; NUKIYAMA, 1966; ROHSENOW; HARTNETT; CHO, 1998; KANDLIKAR, 1999):

- **Natural convection (O–ONB):** The first regime in the boiling curve is natural convection, also known as free convection. In this region, there is no phase change, and monophasic heat exchange happens due to the decrease in density caused by the local increase of the liquid temperature near the heater surface.
- **ONB:** The point known as Onset of Nucleate Boiling (ONB) marks the end of the single-phase, natural convection regime and the start of nucleate boiling.

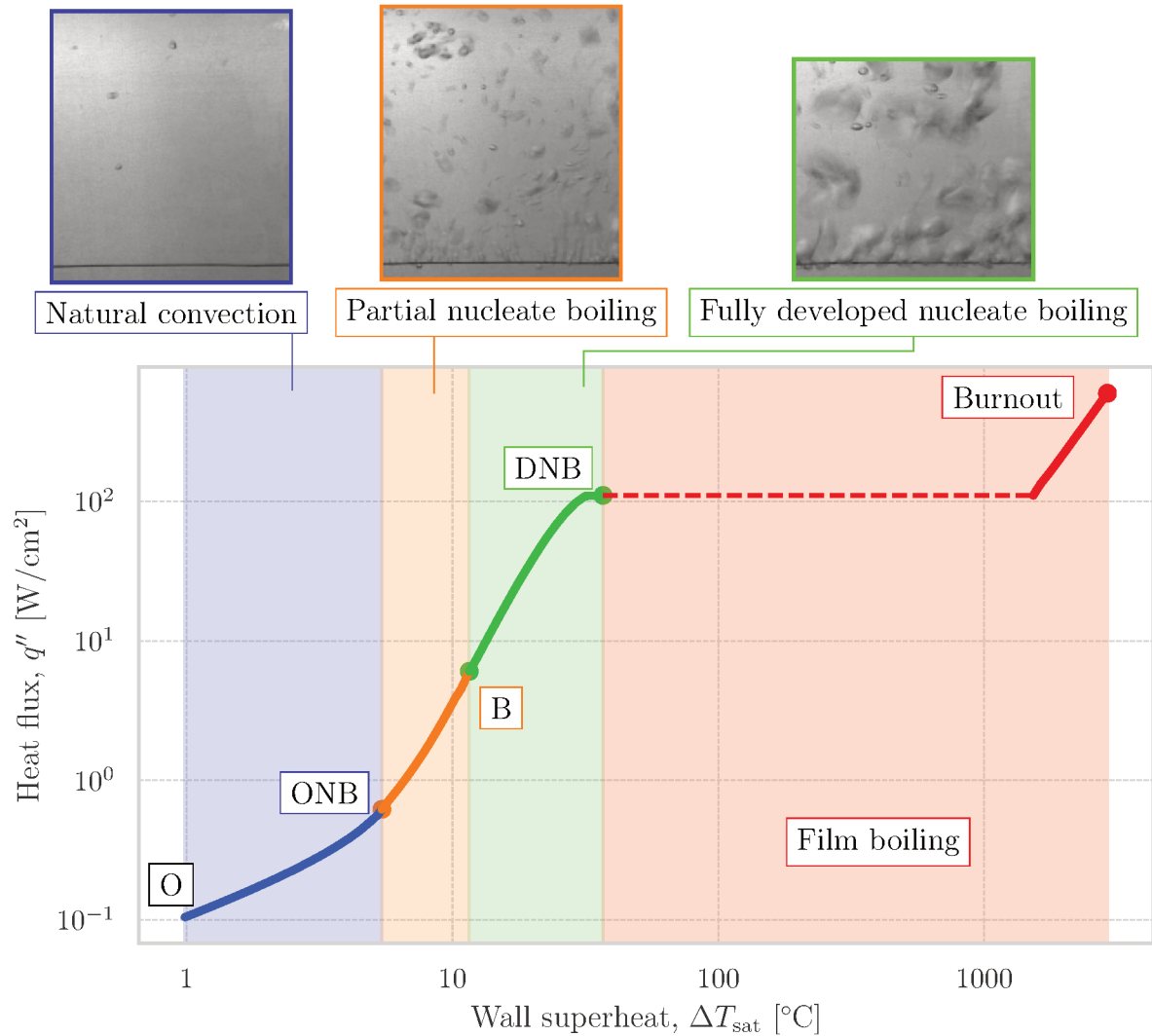


Figure 2 – Boiling curve obtained by controlling the heat flux. Four regimes are highlighted: natural convection, partial nucleate boiling, fully developed nucleate boiling, and film boiling. For the three first regimes, an example frame is presented. Five characteristic points are also shown: the origin (O), the ONB, the transition point (B), the DNB, and the burnout point. The x-y data points utilized to plot the boiling curve were extracted from Çengel and Ghajar (2009).

Cavities on the heater surface act as nucleation sites by trapping vapor. As liquid vaporizes and vapor accumulates in the nucleation site, the liquid-vapor interface is pushed beyond the borders of the cavity, and a bubble nucleates. The bubble grows until the buoyant force that it suffers due to the difference in the liquid and vapor densities overcomes the surface tension force that keeps it attached to the surface. When this happens, the bubble detaches itself from the heater surface and ascends towards the free surface of the liquid.

- **Partial nucleate boiling (ONB–B):** In this region, the main flow regime consists of *isolated bubbles* due to the low density of active nucleation sites; hence this segment of the boiling curve is also known as the *isolated bubbles*

regime. Each bubble grows and detaches itself from the surface independently. In this segment of the boiling curve, higher temperatures activate more nucleation sites and increase the average bubble size. In lower temperatures, bubbles tend to collapse in the liquid phase before reaching the free surface. The detachment of bubbles and their subsequent motion towards the free surface agitates the liquid in the vicinity of the heater surface, enhancing the direct exchange of heat between the liquid and the solid. This mechanism is the primary cause of the increase in the heat transfer coefficient in this region.

- **Fully developed nucleate boiling (B–DNB):** As the heat flux rises, the average bubble size, the frequency of bubble departure, and the density of active nucleation sites increase as well. Consequently, bubbles interfere with each other and coalesce, forming continuous columns and jets of vapor that move in the interior of the liquid and reach the free surface, where the vapor is released into the atmosphere. This regime’s primary heat exchange driver is the combined effect of evaporation and liquid entrainment. The fully developed regime is less sensitive to system parameters compared to partial nucleate boiling.
- **DNB:** The point known as Departure of Nucleate Boiling (DNB) corresponds to the CHF and is characterized by a sharp decrease in the heat transfer coefficient due to the deterioration in the heat transfer mechanisms. This is caused by the formation of a vapor layer between the heater surface and the liquid phase, causing bubbles to grow at the liquid-vapor interface and not on the solid surface. In this situation, heat is transferred primarily via conduction in the vapor film and radiation from the heater surface. The vapor film acts as an insulator, causing the surface temperature to rise sharply.
- **Film boiling (DNB–Burnout):** After the DNB, the decrease in the heat transfer coefficient causes the surface temperature to rise abruptly, possibly exceeding its operating temperature and leading to system failure. This phenomenon is known as the *boiling crisis* and is destructive to many thermal systems, being therefore avoided in various industrial applications.

Other regions and characteristic points can also be observed in the boiling curve. For instance, Kandlikar (1999) explains that, at the ONB, the sudden activation of nucleation sites causes the surface temperature to drop slightly while keeping the heat flux constant. In addition, a region known as *transition boiling* can be reached by carefully manipulating system parameters. Those regions are out of the scope of this work, but more information about them can be found in Rohsenow, Hartnett, and Cho (1998), Kandlikar (1999), Çengel and Ghajar (2009) and Bergman et al. (2011).

Because of the boiling crisis, the CHF is considered the upper limit of the fully developed nucleate boiling regime and the safe operation of the equipment (KANDLIKAR,

1999). Therefore, controlling and, in particular, monitoring boiling processes become mandatory to ensure satisfactory performance and the safe operation of thermal systems. In particular, nuclear reactors require strict temperature control at several points to prevent operational accidents. This monitoring is done mainly through invasive sensors such as Resistance Thermometers (RTDs) and thermocouples (IAEA, 1999; HASHEMIAN; JIANG, 2009). The utilization of such sensors, however, leads to a series of complications. Many factors may impair RTDs' precision or response time, the most prominent ones being their premature failure, calibration errors, and incorrect installation (HASHEMIAN; JIANG, 2009). This way, it is evident a need for developing and improving reliable technologies for monitoring boiling processes quickly and securely.

The general properties of the boiling curve generalize for different conditions of operation. However, the exact shape and positioning of the curve might change, including the ONB and the value of the CHF. Section 2.1.2 discusses in more detail those and other parametric effects.

2.1.2 Parametric effects

Several system parameters impact the pool boiling phenomenon. The observable affected characteristics usually are (a) the boiling curve, including its shape and the position of the characteristic segments described in Section 2.1.1, (b) the density of active nucleation sites, (c) the average bubble formation frequency, and (d) the average departure bubble size (ROHSENOW; HARTNETT; CHO, 1998). The boiling curve is of primary interest to industrial applications since it locates the points of maximal heat transfer and efficiency and the departure from nucleate boiling. The other quantities are also of great interest to visualization-based measurement tools because they are accessible via images, as illustrated in Figure 3.

According to Rohsenow, Hartnett, and Cho (1998), the main system parameters that influence pool boiling are (a) the system pressure; (b) the degree of subcooling; (c) the nature, shape, and surface finishing of the heater surface; (d) the surface inclination; (e) gravity; and (f) the mode of the test, that is, if heat flux increases, decreases, or remains constant during the experimental run. In this work, only the effects of the heater surface are covered in more detail since the other parameters are kept constant during the experiments, as explained in Chapter 3.

Regarding the heater surface effects, many studies investigated the effects that the shape, size, and inclination of the heater surface have on the boiling curve, the CHF, and bubble parameters in the pool boiling phenomenon, such as the bubble departure frequency and diameter. This Section presents some studies about the expected effects that can be visualized in the datasets utilized in this work and which machine learning models can use to make their predictions.

J. H. Kim, You, and Pak (2006) investigated the characteristics of the saturated

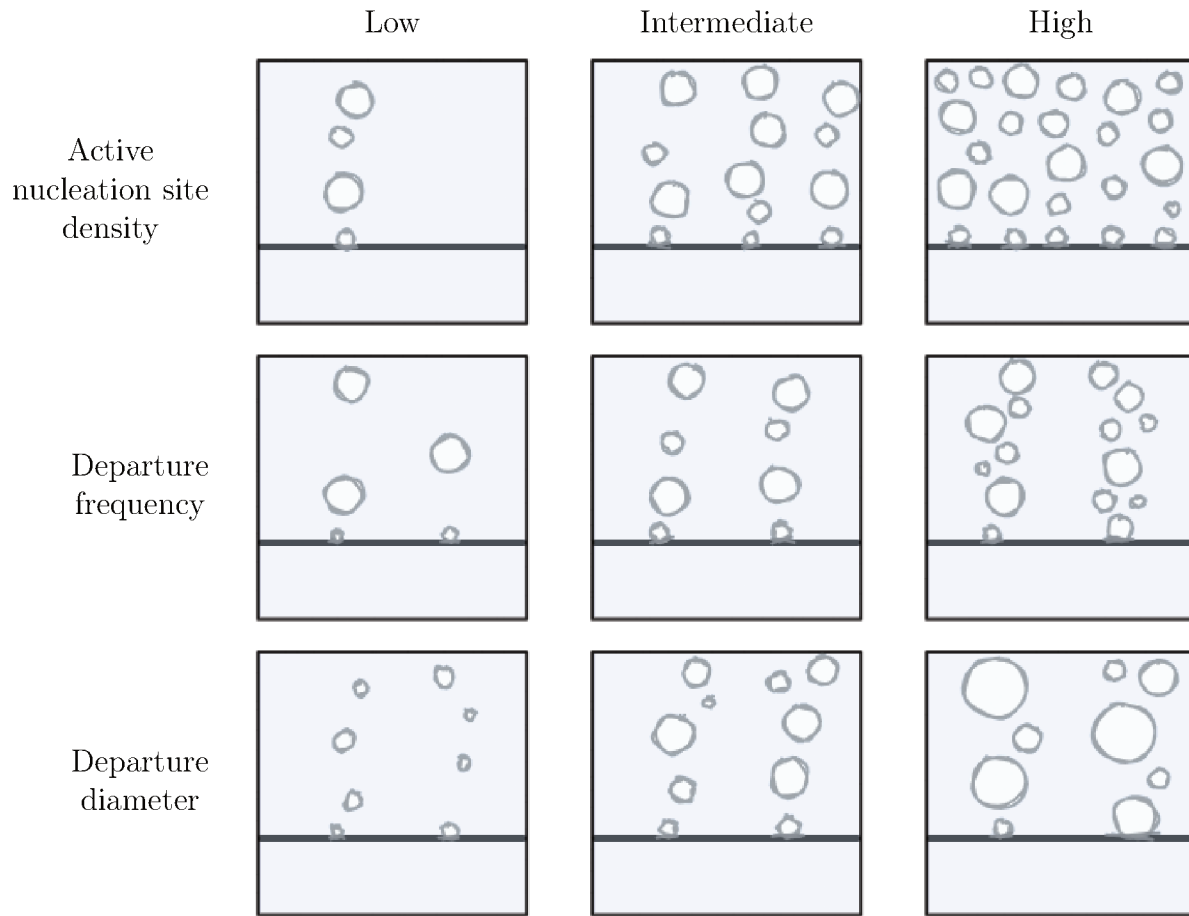


Figure 3 – Visible parametric effects. Each row illustrates how an isolated effect changes the observable behavior of the system.

pool boiling of FC-72 from platinum wires of varying diameters to assess the effects of the heater size on pool boiling. They demonstrated that the wire diameter has little impact on the boiling curve for low heat flux values. On the other hand, smaller wire diameters led to higher heat transfer coefficients in the fully developed nucleate boiling regime, possibly due to a reduction in latent heat contribution. Figure 4 illustrates these effects.

J. H. Kim, You, and Pak (2006) also observed that larger wire diameters result in larger average bubble diameters and a lower frequency of bubble detachment than smaller wires. According to Kandlikar (1999), the two dominant forces that control bubble departure from wires are the buoyancy force and the attachment force between the bubble and the heater surface. Therefore, since thinner wires result in a weaker attaching force to the bubble, smaller buoyancy forces are required for the detachment, which can explain the findings of J. H. Kim, You, and Pak (2006). In addition, J. H. Kim, You, and Pak (2006) attribute the increase in the h to the higher frequency of bubble departure.

Regarding the CHF, Kandlikar (1999) explains that the CHF increases with larger wire diameters for small cylinders such as wires. On the other hand, the opposite tendency is observed for ribbons stuck on plates, with the CHF decreasing on wider ribbons. A possible explanation for this behavior is the different mechanisms according to which

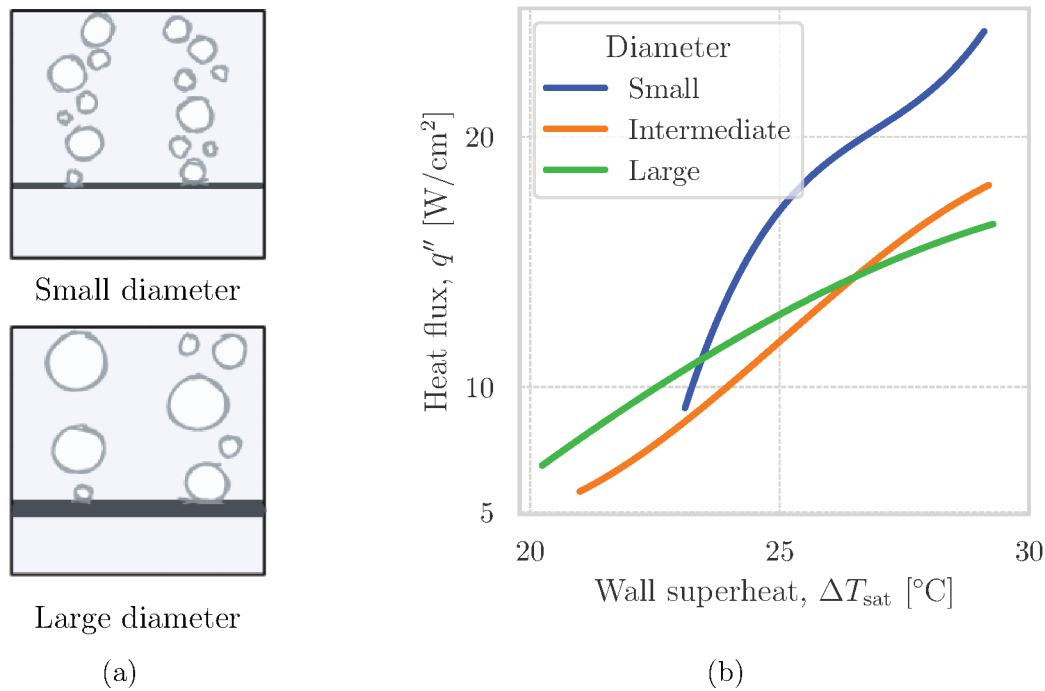


Figure 4 – Boiling curves for small, intermediate, and large wire diameters. To the left (a), illustrations represent wires with small and large diameters. To the right (b), the observed boiling curves obtained from those wires and an intermediate sample are shown. The x-y data points utilized to plot the boiling curves (b) were extracted from J. H. Kim, You, and Pak (2006).

bubbles depart from the heater. On wires, bubbles usually cover the entire cylindrical perimeter, in which case the attaching force and the buoyancy forces are significant. The attaching force is less substantial on ribbons, and the vapor mass departs from the surface subjected mainly to the buoyancy force. Even though this is a possible explanation for why the two cases are different, the mechanisms that affect the CHF are still not completely understood (KOIZUMI et al., 2017).

Koizumi et al. (2017) corroborate those findings by comparing similar studies about the correlations for the CHF on wires and cylinders. Even though there is a disparity between those studies and the data is somewhat scattered, a clear trend was observed: the CHF increases as the wire diameter increases inside the size range in this Thesis's scope.

Kandlikar (1999) and Rohsenow, Hartnett, and Cho (1998) also reviewed many studies that explored the effects of surface inclination on the behavior of pool boiling setups. It was observed that the boiling curve shifts to the left when the surface inclination changes from a horizontal, upward-facing position to a vertical position in the partial nucleate boiling regime, especially at low levels of wall superheat. Inclined surfaces result in a higher heat transfer coefficient at a given heat flux due to the decrease in the wall superheat. This effect can be partially attributed to the movement of bubbles along the heater surface. In contrast, there is little difference in the boiling curve in the fully developed nucleate

boiling regime. Figure 5 illustrates this behavior.

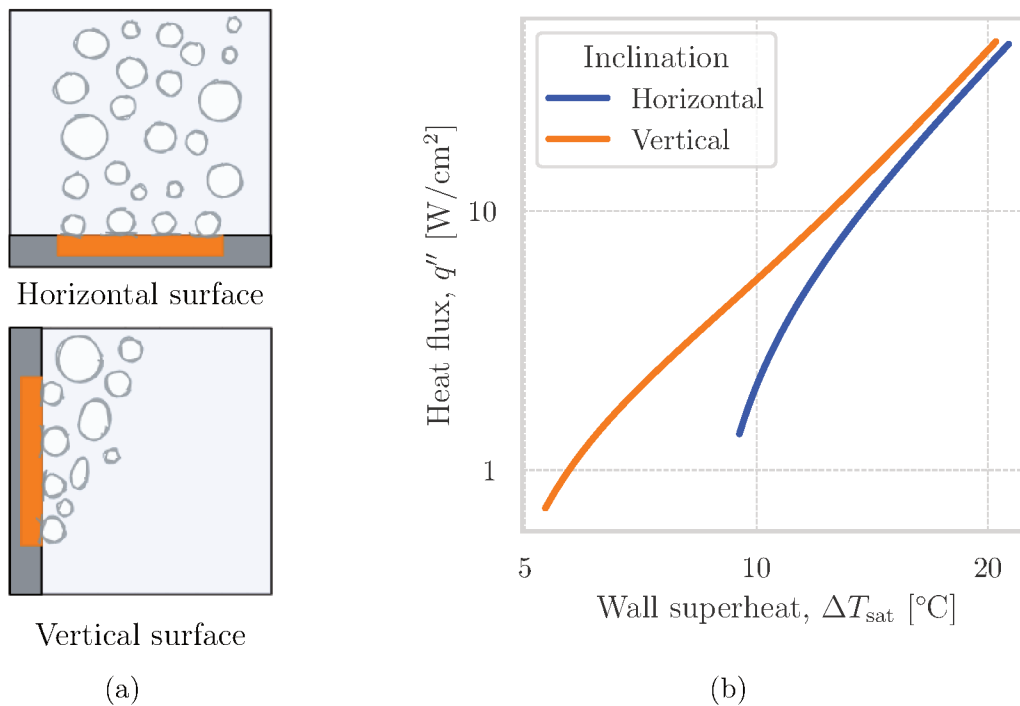


Figure 5 – Boiling curves for horizontal and vertical heater surfaces. To the left (a), illustrations represent a flat heater surface in the horizontal and vertical positions. To the right (b), the observed boiling curves obtained from those positions are presented. The x-y data points utilized to plot the boiling curves (b) were extracted from Rohsenow, Hartnett, and Cho (1998).

Conversely, the CHF is generally lower on inclined surfaces compared to upward-facing, horizontal surfaces (KANDLIKAR, 1999). As Koizumi et al. (2017) explain, this reduction is primarily due to the longer time it takes for vapor masses to move along inclined heater surfaces.

Jung and H. Kim (2016) investigated the effects of surface inclination on the heat transfer rate and bubble parameters in the pool boiling of saturated water from a flat plate. The study demonstrated that, for a fixed wall superheat of 7.5°C , horizontal surfaces exhibit four times less active nucleation sites but twice the bubble departure frequency of vertical surfaces. The increased number of active nucleation sites in vertical surfaces is attributed to the variation in the thermal boundary layer thickness, which activates nucleation sites of different radii along the heater. It was also observed that, on vertical surfaces, bubbles tend to slide up due to buoyancy and coalesce with each other, thus forming one single, large bubble. As a consequence, the average bubble departure diameter increases in inclined surfaces.

Jung and H. Kim (2016) also observed that the boiling curve moves left with increasing surface inclination, meaning that heat flux, and hence the heat transfer coefficient, is larger on vertical surfaces at a given wall superheat. At the superheat $\Delta T_{\text{sat}} = 7.5^{\circ}\text{C}$,

for instance, heat flux was enhanced more than four times on the vertical surface compared to the horizontal case. The increase in the h is likely due to the significant increase in the density of active nucleation sites and the sliding movement of bubbles on the heater surface. These observations are in agreement with the reviews of other studies provided by Rohsenow, Hartnett, and Cho (1998) and Emir et al. (2022).

The effects of the heater size and inclination can hence be summarized as:

- **effects of the heater size:** on wires, as the diameter decreases, the heat transfer coefficient increases at a given temperature. In addition, thinner wires result in smaller departure bubble diameters and a higher frequency of bubble formation. Reducing the heater diameter also reduces the CHF;
- **effects of inclination:** on flat surfaces, it was observed that vertically-oriented surfaces shift the boiling curve to the left, resulting in higher values for the heat flux at a given wall superheat compared to upward-facing surfaces. Additionally, the CHF tends to decrease on inclined surfaces. Regarding bubble parameters, inclined surfaces result in a significantly higher density of active nucleation sites. They also present a lower frequency of bubble formation but a larger average bubble departure diameter.

Finally, as demonstrated by Rohsenow, Hartnett, and Cho (1998), Kandlikar (1999) and Emir et al. (2022), changing the heater surface geometry can alter most of the observable boiling behavior, including the boiling curve and the bubble parameters. Most models and equations depend on assumptions around the surface geometry and become invalid if that changes.

In conclusion, the parametric effects described in this Section justify the choices of heater surfaces in Chapter 3 since both the intrinsic system behavior, represented by the boiling curve and the CHF, and the visible bubble parameters are affected by the heater geometry, size, and inclination.

2.1.3 Numerical correlations

In order to make the project of thermal systems feasible, several authors developed numerical correlations that allow the estimation of the heat flux q'' or the heat transfer coefficient h from system parameters or operating conditions.

According to Bergman et al. (2011) and Çengel and Ghajar (2009), the most widely used correlation for nucleate boiling was developed by Rohsenow (1952), allowing the calculation of the heat flux from the wall superheat ΔT_{sat} and the physical properties of the fluid. The correlation states that

$$q'' = \mu_\ell i_{\ell v} \left[\frac{g(\rho_\ell - \rho_v)}{\sigma} \right]^{1/2} \left(\frac{c_{p\ell} \Delta T_{\text{sat}}}{K i_{\ell v} \text{Pr}_\ell^n} \right)^3, \quad (3)$$

where μ_ℓ is the saturated liquid dynamic viscosity; $i_{\ell v}$ is the specific enthalpy of vaporization; ρ_ℓ and ρ_v are the saturated liquid and vapor massic densities, respectively; σ is the

surface tension; $c_{p\ell}$ is the saturated liquid specific heat at constant pressure; and Pr_ℓ is the saturated liquid Prandtl number. The constants K and n are determined experimentally and depend respectively on the surface-fluid combination and the fluid.

Even though Equation (3) provides a simple, direct correlation between heat flux and wall superheat, it may result in errors of up to 100 % when calculating the heat flux from the wall superheat, or approximately 30 % when calculating the wall superheat from the heat flux (BERGMAN et al., 2011; ÇENGEL; GHAJAR, 2009). Even in the rare case in which all other variables in Equation (3) are known with negligible error, the inherent uncertainty of the equation makes it unfeasible to be used for monitoring and controlling thermal systems. For instance, the uncertainty of 30 % when calculating the wall superheat from the heat flux makes the point of maximum heat transfer coefficient and the DNB indistinguishable for some fluids.

In a review of recent predicting correlations for the heat transfer in pool boiling, Gorenflo et al. (2014) compared nine studies covering 55 fluids under different operating conditions. In the general case, the Heat Atlas (GORENFLO; KENNING, 2010) prediction method was shown to predict heat transfer with the least error, resulting in a Mean Absolute Percentage Error (MAPE) of 9.5 % measured over all 55 fluids. This method corresponds to the correlation

$$\frac{h}{h_{\text{ref}}} = F_{q''} \cdot F_{p^*} \cdot F_\ell \cdot F_s \quad (4)$$

where $F_{q''}$ represents the dependency of the heat transfer coefficient h on the heat flux q'' ; F_{p^*} is the dependency on the reduced pressure p^* ^[1]; and F_ℓ and F_s are functions of the thermophysical properties of the working fluid and the heater surface, respectively. The quantity h_{ref} is the *reference* heat transfer coefficient, a reference value used for all fluids. Even though the Heat Atlas (GORENFLO; KENNING, 2010) correlation shows promising results, it requires specific data fitting to determine F_ℓ and F_s . Despite having a smaller associated error than Equation (3), this uncertainty can still be prohibitively high for real-time estimation and control of the boiling process.

Specifically for the pool boiling of water, Gorenflo et al. (2014) reported that, among the correlations they analyzed, the one by Yagov (2009) performed the best, achieving 4.82 % of error for predicting the heat transfer coefficient for the reference experimental data. The correlation states that

$$h = 3.43 \times 10^{-4} \left(1 + \frac{i_{\ell v} \Delta T_{\text{sat}}}{2RT_{\text{sat}}^2} \right) \frac{\lambda_\ell^2 \Delta T_{\text{sat}}^3}{\nu_\ell \sigma T_{\text{sat}}} \left[1 + (1 + 800B)^{1/2} + 400B \right] \quad (5)$$

where

$$B = \frac{i_{\ell v} (\nu_\ell \rho_v)^{3/2}}{\sigma (\lambda_\ell T_{\text{sat}})^{1/2}}, \quad (6)$$

^[1] The *reduced pressure* p^* is defined as $p^* = p/p_c$ where p is the local pressure and p_c is the *critical pressure* of the fluid, that is, the pressure at its critical point.

R is the specific gas constant, λ is the thermal conductivity and ν is the kinematic viscosity.

Despite having a relatively low error associated, Equation (5) requires measuring ΔT_{sat} , a process which may itself bring high uncertainty to the prediction. Additionally, as noted by Gorenflo et al. (2014), those correlations take little, if any, information about the heater surface into account, thus being constrained to the surface for which they were developed. Understanding the effects of the heater surface and incorporating them into the correlations is the top priority for future studies.

Emir et al. (2022) reviewed more than 30 correlations for estimating the CHF in various conditions, such as different working fluids, system pressures, subcooling temperatures, orientations, and surfaces. The correlations behave relatively well and provide crucial insight into the understanding of pool boiling systems. However, they are typically associated with uncertainties of around 25% or above. One of the correlations covered in that review was fine-tuned with more data points, and the authors were able to reduce its associated uncertainty to 7.5%, representing a substantial improvement. However, even though the correlations analyzed behave relatively well within their applicability range, they cannot be extrapolated to other operating conditions.

In summary, as detailed in this Section, empirical and semi-empirical correlations allow the calculation of heat transfer parameters such as the heat flux, the heat transfer coefficient, and the CHF from other quantities such as operating conditions and fluid properties. However, those correlations are usually associated with uncertainties or errors that limit their applicability. In addition, generic correlations that cover multiple, different use cases are frequently associated with higher uncertainties, which poses a trade-off between their applicability and the confidence around their predictions.

2.1.4 Mechanistic models

Alternatively to the empirical or semi-empirical numerical correlations, described in Section 2.1.3, mechanistic models allow the calculation of heat transfer parameters from physical, observable inputs, such as the average bubble departure diameter and the number of active nucleation sites. Those models are more similar to visualization-based machine learning models (detailed in Section 2.2) since the same inputs are directly or indirectly accessible to them.

Based on mechanistic analyses, Mikić and Rohsenow (1969) developed an expression for calculating the heat flux q'' or the heat transfer coefficient h from visible and measurable quantities such as the density of active nucleation sites, N_a ; the average bubble diameter at departure, D_d ; and the average frequency of bubble formation, f . Their model considered the contribution of transient heat conduction in the neighborhood of the active nucleation sites and of natural convection in the inactive regions as the two governing sources of heat transfer in the nucleate regime. Because of the clear separation between the different contributors to the heat flux, this model is known as the *heat flux partitioning model*.

Their model was later expanded by Judd and Hwang (1976) to account for evaporation in the microlayer beneath the bubbles, resulting in the expression

$$h = \left[\frac{K_{\text{fit}}^2}{2} C_\ell N_a D_d^2 \sqrt{f} + \left(1 - \pi \frac{K_{\text{fit}}^2}{2} N_a D_d^2 \right) h_{\text{NC}} + \frac{\pi}{4} N_a D_d^2 h_{\text{EV}} \right] \quad (7)$$

where K_{fit} is a numerical constant obtained from fitting experimental data and C_ℓ is a function of the saturated liquid properties. The heat transfer coefficients h_{NC} and h_{EV} represent the coefficient obtained solely from natural convection and the one due to microlayer evaporation, respectively. The comparison of this model with experimental data showed good agreement in behavior, but the deviation was considerably high.

According to M. Kim and S. J. Kim (2020), other studies further improved Equation (7). However, errors were still significantly high, particularly close to the DNB, where bubble coalescence becomes prominent. To address this shortcoming, they proposed a new mechanistic model incorporating the effects of bubble coalescence in the pool boiling from a flat surface. According to their model,

$$h = \frac{\lambda_\ell}{\sqrt{\pi \alpha_\ell t_w}} \left\{ 1 + \frac{\pi}{4} D_d^2 N_a \left[C_{A,1} K_A^2 + C_{A,2} \left(\frac{5}{2} \sqrt{\frac{3\pi}{\text{Pr}_\ell}} - 2 \right) \sin^2 \theta \right] \right\}, \quad (8)$$

where α is the thermal diffusivity; t_w is the bubble waiting time for departure; K_A is an adjusted constant known as the *influence area factor*, defined as the ratio of the diameter of the quenching region to the diameter of the evaporation region; θ is the contact angle between the heater surface and the bubble interface; and $C_{A,1}$ and $C_{A,2}$ are area correction factors which can be calculated from the distance between adjacent bubbles and their departure diameter.

M. Kim and S. J. Kim (2020) evaluated their proposed model on a saturated water dataset and measured the Root Mean Squared Error (RMSE) between the modeled boiling curve and experimental data. The RMSE reduced from 239 kW/m², obtained using the heat flux partitioning model, to 20 kW/m² using the new model, an improvement of more than 90%.

Despite the promising results obtained with mechanistic models and numerical correlations, those approaches are limited to the specific use cases for which they were designed. In addition, they depend on input quantities that need to be measured, a process that increases the global uncertainty around the final prediction. Furthermore, for real-time measurement and control of thermal systems, that process may even be unfeasible depending on the response time of the measurement systems. Therefore, recent studies have attempted several alternative approaches for the measurement of heat transfer parameters in order to overcome those limitations, such as the ones presented in Section 2.3.

2.2 MACHINE LEARNING

The advances in computer hardware and software in the last decades made feasible the application of progressively more efficient and robust computational algorithms in a multitude of situations. Among such new technologies, Machine Learning (ML) rises as one of the most powerful tools available for solving data-centric problems. Algorithms that learn with their experience have been successfully applied in the last decade in various areas such as the identification of risk factors for cancer (FRIEDMAN; HASTIE; TIBSHIRANI, 2001), stock market forecasting (JAMES et al., 2013), the recognition of objects and faces in images (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; HE et al., 2016; SANDLER et al., 2018), and many others (GOODFELLOW; BENGIO; COURVILLE, 2016).

Machine Learning can be defined as the improvement of a machine’s performance in executing a task as it gains experience (MITCHELL, 1997). According to Goodfellow, Bengio, and Courville (2016), learning can be categorized as *supervised* or *unsupervised*, the former of which is the focus of this work. In supervised learning, models consume *datasets* in the form $\mathcal{D} = \left\{ \left(\mathbf{x}^{(i)}, y^{(i)} \right) \right\}_{i=1}^m$, where each $\mathbf{x}^{(i)}$ is an *input* called a *feature vector*, and $y^{(i)}$ is the corresponding desired *output*. The two most popular examples of supervised learning are *regression* and *classification* problems. In regression problems, each $y^{(i)} \in \mathbb{R}$, known as a *target*, is assumed to belong to a bounded, continuous real interval. On the other hand, in classification problems, each example belongs to a specific class; in this situation, each $y^{(i)}$ is called a *label* and is an element of the set $\{1, \dots, C\}$ where C is the number of classes in the problem.

Machine Learning fundamentally differs from Optimization in its goal. Optimization aims to minimize an error metric computed over a training set $\mathcal{D}_{\text{train}}$. On the other hand, in ML, although there is still an optimization problem over the training set to be solved, the goal is to minimize the generalization error, that is, the expected error over a new, unseen test set $\mathcal{D}_{\text{test}}$. In order to estimate the generalization error and use this estimate to choose or calibrate different models, a validation set \mathcal{D}_{val} is defined as a subset of the training data. With this setup, the training set $\mathcal{D}_{\text{train}}$ is utilized to learn the model’s *parameters*, that is, to find an optimal function in a parameterized class. In turn, the validation set \mathcal{D}_{val} is employed to choose the model’s *hyperparameters*, non-trainable parameters that control the learning algorithm. Finally, the model’s performance is measured on the test set $\mathcal{D}_{\text{test}}$ (GOODFELLOW; BENGIO; COURVILLE, 2016).

A standard definition in the literature is the differentiation between a model’s parameters and hyperparameters. Parameters are the variables that specify a function from the parametrized class associated with the model. In deep learning models, described in Section 2.2.1, those are the trainable weights. In contrast, hyperparameters are the non-trainable variables that configure the training algorithm or define the parametrized class. The number of layers and units in deep learning models, the Adam optimizer learning

rate (KINGMA; BA, 2014), and the dropout rate (SRIVASTAVA et al., 2014) are examples of hyperparameters.

For evaluations on the test set to be statistically significant, unbiased estimates for the generalization error, it is crucial that test examples are never shared with the training and validation sets. Similarly, performance metrics evaluated on the test set cannot be used to tune training hyperparameters since that would introduce bias. Consequently, test metrics are only utilized in this work for communication and reporting purposes, not for designing machine learning models. Because of this limitation, the validation set is employed to estimate the generalization error when training models and tuning hyperparameters.

Ideally, a model learns the relationship between the examples in the training set and thus generates a correct representation for them or *fits* the data. If this does not happen, and the training error is significantly high, the model is said to *underfit* the data. In general, models underfit when their *hypothesis space*, the set of functions from which the learning algorithm selects the solution, does not represent the training data. Some standard practices to prevent underfitting include: changing the type of model, for instance, replacing a polynomial regressor with a neural network; increasing the number of learnable parameters, for instance, by increasing the number of units in a neural network; changing the model's architecture, e.g., by adding convolutional layers to a neural network, as explained in Section 2.2.2; or tuning the model's hyperparameters (GOODFELLOW; BENGIO; COURVILLE, 2016).

In contrast, a model *overfits* the data when the generalization error is considerably higher than the training error. Intuitively, a model overfits when it learns not only the intrinsic properties of the distribution it models but also the *noise* present in the data samples (MURPHY, 2012), thus memorizing aspects of the training set that are not present in the test set (GOODFELLOW; BENGIO; COURVILLE, 2016). The two most common techniques for preventing overfitting are: increasing the training set size and applying regularization techniques, such as ℓ_2 penalization or *dropout* (SRIVASTAVA et al., 2014).

2.2.1 Deep learning

Many ML techniques were developed in the past, aiming to model progressively more complex functions. Among the most modern technologies, neural networks stand out as powerful tools due to their high versatility, performance, and scalability. According to Goodfellow, Bengio, and Courville (2016), deep neural networks, containing multiple layers, succeed in a wide range of tasks because they rely on the general principle of learning multiple levels of composition.

Neural networks approximate any continuous, multivariate function with arbitrary precision as long as an adequate architecture is chosen and enough computation time is given (GOODFELLOW; BENGIO; COURVILLE, 2016). In particular, CNNs are widely

applied to computer vision problems by utilizing convolutional operations to extract features from image datasets, which are typically highly dimensional (MURPHY, 2012). For simplicity, this Section introduces MLPs, whereas CNNs are described in depth in Section 2.2.2.

An MLP consists of the composition of multiple layers of independent learning *units*. An L -layers MLP can be illustrated as in Figure 6a. The first layer, corresponding to the layer index $\ell = 0$, is known as the *input layer* and corresponds to the feature vector $\mathbf{x} \in \mathbb{R}^n$. The last layer, corresponding to $\ell = L$, is called the *output layer* and results in the prediction $\hat{\mathbf{y}} \in \mathbb{R}^p$. For $0 < \ell < L$, layers are known as *hidden layers* since they are internal to the network (GOODFELLOW; BENGIO; COURVILLE, 2016).

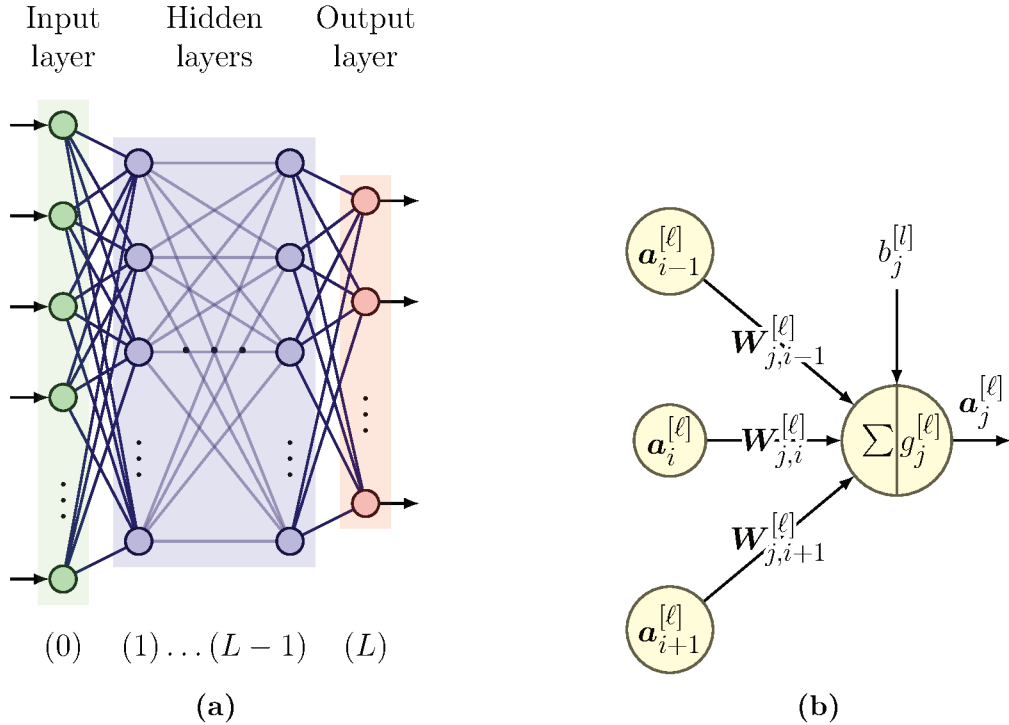


Figure 6 – Illustrations of (a) an MLP and (b) a unit.

Each layer ℓ is composed of n_ℓ units, with $n_0 = n$ and $n_L = p$. Each unit j from layer ℓ connects to all $n_{\ell-1}$ units from the previous layer and receives as inputs the activations $\mathbf{a}^{[\ell-1]} = (\mathbf{a}_1^{[\ell-1]}, \dots, \mathbf{a}_{n_{\ell-1}}^{[\ell-1]})$ produced by them. Unit j then applies a linear transformation to the previous layer's activations, computing $\mathbf{z}_j^{[\ell]} = \mathbf{W}_j^{[\ell]} \mathbf{a}^{[\ell-1]} + \mathbf{b}_j^{[\ell]}$, where $\mathbf{b}_j^{[\ell]}$ is a *bias* term. Finally, the activation for unit j is obtained by means of the application of a non-linear function $g^{[\ell]}$ to the previously calculated linear combination: $\mathbf{a}_j^{[\ell]} = g^{[\ell]}(\mathbf{z}_j^{[\ell]})$. This unit is illustrated in Figure 6b.

This way, an MLP defines a chain of transformations

$$\mathbf{a}^{[0]} = \mathbf{x}, \quad \mathbf{z}^{[\ell]} = \mathbf{W}^{[\ell]} \mathbf{a}^{[\ell-1]} + \mathbf{b}^{[\ell]}, \quad \mathbf{a}^{[\ell]} = g^{[\ell]}(\mathbf{z}^{[\ell]}), \quad \hat{\mathbf{y}} = \mathbf{a}^{[L]}, \quad (9)$$

where $\mathbf{W}^{[\ell]} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $\mathbf{b}^{[\ell]} \in \mathbb{R}^{n_\ell}$. The function $g^{[\ell]}$ is known as the *activation function* for layer ℓ , and is applied to $\mathbf{z}^{[\ell]}$ element-wise, resulting in the *activation* $\mathbf{a}^{[\ell]}$. With this, the hidden layers learn non-linear correlations between their inputs in what is known as *automatic feature extraction* (MURPHY, 2012). Because of the high connectivity between their units, MLP layers are known as *densely connected layers*, or simply *dense layers*. Note that each dense layer ℓ contains $n_\ell \times n_{\ell-1} + n_\ell$ independent parameters, corresponding to the elements in $\mathbf{W}^{[\ell]}$ and $\mathbf{b}^{[\ell]}$.

The activation function $g^{[\ell]}$ allows models to learn non-linear relationships between their inputs and outputs. As such, choosing the activation function is an important step for designing training pipelines. According to Goodfellow, Bengio, and Courville (2016), the most commonly utilized activation function in CNNs is the ReLU, defined as

$$\text{ReLU}(x) = \max\{0, x\}. \quad (10)$$

Details around this and other activation functions can be found in Murphy (2012) and Goodfellow, Bengio, and Courville (2016).

In order to train an MLP, a *cost function* J must be chosen and minimized. The simplest approach is to calculate the cost function from a *loss function* computed over the entire dataset:

$$J = \frac{1}{m} \sum_{i=1}^m L(\mathbf{y}^{(i)}, \hat{\mathbf{y}}^{(i)}). \quad (11)$$

According to Chollett (2018), in regression problems, it is common to choose the quadratic error as the loss function, in which case the cost function equals the Mean Squared Error (MSE) explained in further detail in Section 2.2.3:

$$J = \frac{1}{m} \sum_{i=1}^m \left(y^{(i)} - \hat{y}^{(i)} \right)^2 = \text{MSE}. \quad (12)$$

Additionally, Equation (11) is frequently adjusted to include regularization terms.

The training stage involves finding the parameters $\mathbf{W}^{[\ell]}$ and $\mathbf{b}^{[\ell]}$ that minimize the cost function. To that end, gradient-based algorithms, such as Stochastic Gradient Descent (SGD) and Adam (KINGMA; BA, 2014), were developed. In this context, the *backpropagation* method rises as an efficient algorithm for calculating the cost function gradient via the application of the chain rule to Equation (9). According to Goodfellow, Bengio, and Courville (2016), the generalization capability of neural networks derives from utilizing non-linear activation functions and the backpropagation method.

There are many other aspects of training deep models, such as the initialization of weights, the choice of learning rates and momenta and learning rate decay, whose detailed explanation can be found in Goodfellow, Bengio, and Courville (2016), Murphy (2012), Chollett (2018), Bishop (2006) and Friedman, Hastie, and Tibshirani (2001).

2.2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a type of neural network designed to efficiently process and learn from data organized in a grid-like format. This data can be arranged in 1D grids, as in time series, or 2D or 3D grids, as in grayscale or colored image data. This Section explains the bi-dimensional case in more detail.

CNNs contain specialized *convolution* layers that apply matrix operations, or filters, to groups of neighbor pixels. Each filter is represented by a real matrix called a *kernel* and is convoluted with its input by computing the dot-product with its windowed submatrices, as illustrated in Figure 7 (GOODFELLOW; BENGIO; COURVILLE, 2016).

$$\begin{array}{ccc}
 \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} & * & \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 3 & 4 & 1 \\ 1 & 2 & 4 & 3 & 3 \\ 1 & 2 & 3 & 4 & 1 \\ 1 & 3 & 3 & 1 & 1 \\ 3 & 3 & 1 & 1 & 0 \end{bmatrix} \\
 \mathbf{I} & & \mathbf{K} \qquad \qquad \mathbf{I} * \mathbf{K}
 \end{array}$$

Figure 7 – Demonstration of the application of a convolutional layer filter. Each element in the output matrix $\mathbf{I} * \mathbf{K}$ is calculated as the dot-product between the corresponding window in the input matrix \mathbf{I} and the kernel \mathbf{K} . The values in the matrices were extracted from Neutelings (2022).

In convolutional layers, each kernel contains learnable parameters which are tuned during the training stage. For instance, a 2×2 kernel \mathbf{K} can be written as

$$\mathbf{K} = \begin{bmatrix} k_{1,1} & k_{1,2} \\ k_{2,1} & k_{2,2} \end{bmatrix}, \tag{13}$$

where $k_{1,1}$, $k_{1,2}$, $k_{2,1}$ and $k_{2,2}$ are learnable parameters. In addition, a bias term $b_{\mathbf{K}} \in \mathbb{R}$ is usually included so that each convolutional filter defines a transformation

$$\text{CONV}(\mathbf{I}, \mathbf{K}) = \mathbf{I} * \mathbf{K} + b_{\mathbf{K}}. \tag{14}$$

Each convolutional layer contains multiple filters, typically 32 or 64 (CHOLLETT, 2018). A convolutional layer of N filters of shape $H \times W$ thus adds $N(HW + 1)$ trainable weights to the model, including the bias term. Goodfellow, Bengio, and Courville (2016) and Chollett (2018) additionally describe *padding* and non-unitary *strides*, which are also important aspects of convolutional layers.

According to Chollett (2018), the main difference between dense layers, described in Section 2.2.1, and convolutional layers is that the first group learns *global* patterns in their input feature space, whereas the second group learns *local* patterns.

CNNs are of great interest to the modeling of image data due to four main characteristics (GOODFELLOW; BENGIO; COURVILLE, 2016; CHOLLETT, 2018):

- **sparse connectivity:** in contrast to dense layers in which every input pixel connects to every output pixel, in convolutional layers, each output is connected only to the kernel parameters, which is normally several orders of magnitude smaller than the input images. Therefore, the number of connections, and hence matrix operations, is significantly reduced in convolutional layers, reducing their memory footprint and runtime costs compared to dense layers.
- **parameter sharing:** because each filter is applied to every window of the input image without changing its parameters, those parameters are *shared* between the input windows. In contrast, dense layers contain a different set of trainable weights for each input pixel. As a consequence, the storage requirements of convolutional layers tend to be much lower than those of dense layers.
- **translation-equivariance:** since the kernels are applied to all windows of the input image, the patterns they learn can be identified anywhere in the input, regardless of where they appear. If an object is translated in the input image, then its representation in the output is also equivalently translated without changing its contents. In contrast, dense layers would require intensive retraining to detect the same object in a different location. This makes convolutional networks exceptionally efficient when processing time series and image data. In particular, convolutional layers require fewer training samples to learn generalized representations since they are naturally equivariant to translation.
- **spatial hierarchy learning via composition:** CNNs learn spatial hierarchies of patterns by stacking, or composing, convolutional layers. In this scenario, the first layers, closer to the input, learn smaller and simpler local patterns, such as edge detection; the following layers can then learn more extensive, complex, and abstract concepts, such as object detection.

Typically, convolutional layers are followed by two additional layers (GOODFELLOW; BENGIO; COURVILLE, 2016). The first layer is an activation layer where the output of the convolution stage is activated by a non-linear activation function u , similarly to Equation (9). The standard choice for the activation layer following convolutional layers is the ReLU activation. The second layer is a *pooling layer* responsible for reducing data dimensionality and providing local *invariance*.

The pooling layer applies a pooling function to windows of the activated output from the convolutional layer. By definition, a pooling function reduces its input window to a single value by calculating a summary statistic. The most popular pooling function is *max-pooling*, which chooses the maximum value in the window, as illustrated in Figure 8. Max-pooling is conceptually similar to a convolution, except that it uses a hard-coded **max** operation instead of learning a linear transformation (CHOLLETT, 2018).

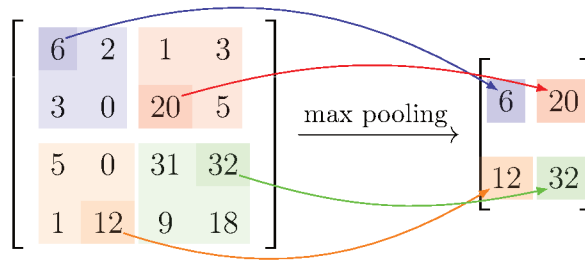


Figure 8 – Demonstration of the application of a 2×2 max-pooling operation. Each output pixel is the maximum of the corresponding 2×2 window in the input image.

The primary purpose of using max-pooling is to reduce the number of feature map coefficients that need to be processed. The dimensionality reduction is essential since each convolutional layer of N filters with unitary strides outputs N images of nearly, or exactly, the same size as the input^[2], as illustrated in Figure 7. Furthermore, pooling layers create spatial filter hierarchies by making successive convolution layers examine larger windows relative to the original input. This helps improve computational efficiency and extract more meaningful features from the input data (GOODFELLOW; BENGIO; COURVILLE, 2016; CHOLLETT, 2018).

Max-pooling tends to work better than alternative pooling options, such as average pooling, because the preceding convolutional layers tend to encode the presence of abstract patterns and concepts into their output. By picking the maximum activation in a neighborhood, the max-pooling layer instructs the model to look at the maximal presence pattern in the feature map. This is normally more useful than searching for the average presence of features, as the average pooling does (CHOLLETT, 2018).

Finally, CNNs are typically topped by a sequence of dense layers, described in Section 2.2.1. This way, the convolutional layers act as automatic feature extractors, transforming the input image into an abstract representation for the dense layers to consume. Figure 9 illustrates a CNN from input to output.

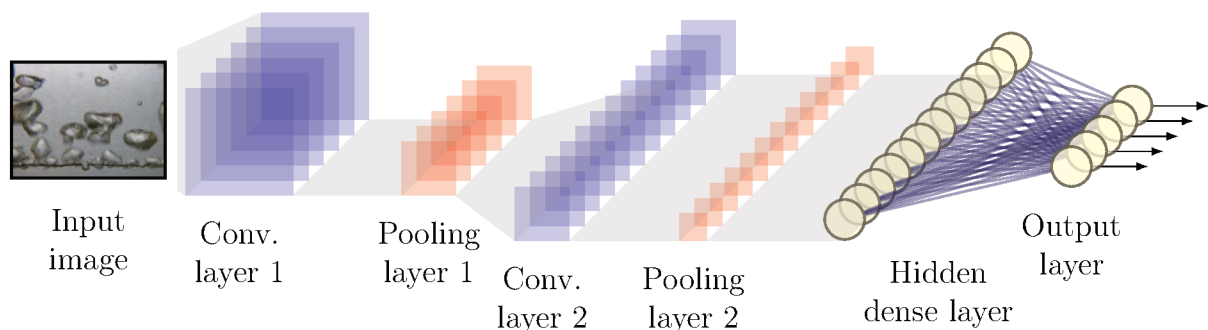


Figure 9 – Illustration of a convolutional neural network containing two convolutional layers, each followed by a pooling layer. The convolutional layers are followed by a hidden, dense layer and topped by the output layer.

^[2] The exact shape of the outputs of a convolutional layer depend on the chosen padding algorithm.

2.2.3 Performance metrics

Several metrics can be used to assess the performance of trained models. In regression problems, the most commonly applied metrics are error metrics that evaluate the average distance between the data points and the predicted values.

In this work, models were analyzed using the MSE, the RMSE, the Mean Absolute Error (MAE), the MAPE and the R^2 score, respectively defined as

$$\begin{aligned} \text{MSE} &= \frac{1}{m} \sum_{i=1}^m \left(y^{(i)} - \hat{y}^{(i)} \right)^2, & \text{RMSE} &= \sqrt{\text{MSE}}, & \text{MAE} &= \frac{1}{m} \sum_{i=1}^m \left| y^{(i)} - \hat{y}^{(i)} \right|, \\ \text{MAPE} &= \frac{1}{m} \sum_{i=1}^m \left| \frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}} \right|, & R^2 &= 1 - \frac{\sum_{i=1}^m \left(y^{(i)} - \hat{y}^{(i)} \right)^2}{\sum_{i=1}^m \left(y^{(i)} - \bar{y} \right)^2} \end{aligned} \quad (15)$$

where $\bar{y} = \frac{1}{m} \sum_{i=1}^m y^{(i)}$ is the average of the targets.

The MSE, RMSE, MAE, and MAPE equal zero only for perfect predictors, for which all predictions match their respective true values; they are positive otherwise, with the MSE and RMSE being more sensitive to outliers than the MAE. The MSE is commonly used as the *loss function* during the training of regression models (CHOLLETT, 2018), as well as a measure of the overall expected deviation on the validation set.

Finally, the R^2 score evaluates models by comparing them with a trivial baseline model. A perfect predictor would have a R^2 score of 1 because, in this case, $\hat{y}^{(i)} = y^{(i)}$. On the other hand, a model scoring 0 would have the same performance as the *trivial model*, the constant mean predictor for which every prediction is constant and equal to the average target, i.e. $\hat{y}^{(i)} = \bar{y}$; such models are normally discarded since they do not outperform the trivial model (ABADI et al., 2015c).

In addition, the MAPE allows comparing models or datasets that operate on different scales. However, it is important to note that the MAPE is asymmetric: it incentivizes predictions with lower absolute value since low predictions are at most 100% wrong, whereas high predictions' errors are unbounded (TOFALLIS, 2015). This property may be especially problematic when the MAPE is employed as the objective metric for selecting models to serve as measurement tools, as they will tend to underestimate the measured variable, potentially incorrectly labeling a physical system to be in a safe operating condition when it is close to failure.

Other metrics not mentioned in this Section are also helpful in different contexts. Goodfellow, Bengio, and Courville (2016), Chollett (2018) and Abadi et al. (2015a) provide more detailed information about them and their use cases.

The MSE is employed as both the training loss and the main performance metric in this work. The other metrics are provided to ease the comparison with other studies and provide further insight into the meaning of results. The search for performance

metrics is still an active field of research, and more studies are necessary to find the best solution for measuring the performance of heat transfer prediction models (HUGHES; KINI; GARIMELLA, 2021).

2.2.4 Automated machine learning

Given a fixed model architecture, training a deep learning model consists of minimizing the loss function defined in Equation (11) by optimizing the model's parameters. However, this process assumes that many preliminary decisions are made, including the choice of the types of layers in the model (e.g., convolutional or dense layers) and the optimizer algorithm (such as SGD or Adam (KINGMA; BA, 2014)) as well as their associated hyperparameters, for instance, the number of filters in a convolutional layer or the learning rate in the Adam optimizer. Typically, this decision is made by human ML practitioners according to their previous experiences (SONG; JIN, Haifeng; HU, 2022), and this is how the studies covered in Section 2.3 designed their models.

According to Song, Haifeng Jin, and Hu (2022), the human-driven process can be ineffective or suboptimal due to various factors, such as the cost of acquiring the necessary knowledge and experience about ML, the complexity of the implementation process, and the gap between theory and practice. Good and thorough knowledge is required to formulate a problem as an ML problem, select suitable algorithms and preprocess data, which may take significant time to research and learn. Implementation and debugging can also be complex, especially with advanced algorithms. In addition, the performance of an ML model often depends on the data it is trained on, which can be noisy and difficult to interpret, clean, and control, leading to an empirical tuning process that relies on trial and error. Even experienced practitioners may face difficulties in certain cases.

In this context, Automated Machine Learning (AutoML) rises as an excellent tool for overcoming those limitations. AutoML consists of automating the search and evaluation of different ML algorithms, including the choice of architecture, layer size, optimizer, and regularization techniques. Song, Haifeng Jin, and Hu (2022) delineate the three core components of an AutoML procedure:

- **Search space:** the set of hyperparameters the AutoML algorithm can choose from to build and evaluate models. For instance, the search space for the number of filters in convolutional layers may be specified as the finite set $\{32, 64\}$. Other hyperparameters, such as the optimizer learning rate, must be sampled from a continuous real interval, e.g. $[1 \times 10^{-5}, 1]$. In principle, the search space can be as large as necessary to achieve the desired performance at the cost of demanding more time for the search algorithm to run. Therefore, a process of *search space design* is mandatory to reduce the computation required to perform the search. This process involves pruning the search space by incorporating knowledge or task requirements. An example of search space design can be

found in Section 4.5.

- **Search strategy:** the strategy to select an optimal set of hyperparameters from the search space. The most trivial search strategy is known as *grid search* and consists of testing all possible combinations of hyperparameters. Even though grid search is guaranteed to find the best possible hyperparameters from the search space, it is often prohibitively expensive since it requires the entire training pipeline to be executed once per set of hyperparameters. Therefore, more cost-effective strategies are usually employed to reduce the search time and computational cost. The default strategy in AutoKeras (JIN, Haifeng; SONG; HU, 2019) is the *greedy* algorithm, which separates hyperparameters into different categories, such as preprocessing-related, architecture-related, and optimizer-related hyperparameters. At each trial, the algorithm randomly selects a set of hyperparameters from one category while keeping the hyperparameters from other categories equal to the best ones ever observed. Other strategies, such as Bayesian optimization (BERGSTRA et al., 2011; HUTTER; HOOS; LEYTON-BROWN, 2011; SNOEK; LAROCHELLE; ADAMS, 2012) or Hyperband (LI et al., 2016), are also available from AutoKeras (JIN, Haifeng; SONG; HU, 2019).
- **Performance evaluation strategy:** the strategy to evaluate the selected hyperparameters. Generally, this is done by instantiating a model with the selected hyperparameters and computing performance metrics. Thus, the best model is the one with better performance according to those metrics.

2.3 MACHINE LEARNING-BASED ANALYSIS OF PHASE-CHANGE

Given the need for better heat transfer prediction tools, as pointed out in Section 2.1, several studies employed visualization-based machine learning models to quantify heat transfer or classify regimes in the context of the boiling process. This Section presents the most relevant and closely-related works found in the literature.

Hobold and da Silva (2018b) demonstrated the application of machine learning techniques to the classification of pool boiling regimes from a nichrome wire. Two types of classifiers were trained: SVMs and shallow MLPs^[3]. Their dataset comprised images obtained by a conventional camera with a frame acquisition rate of 30 fps and a resolution of 1280×720 pixels (720 p). A detailed review of their experimental setup can be found in Chapter 3 since the present work employed the same experimental bench. Due to the relatively high number of features in their images, they applied dimensionality reduction techniques. In order to do so, colored images were converted to grayscale as their first preprocessing step, representing a threefold reduction in the number of pixels. This pre-

^[3] A *shallow* MLP contains, by definition, a single hidden layer.

processing step can be found in more detail in Section 4.2.2.3. The authors subsequently applied a downscaling transformation, described in Section 4.2.2.4, to reduce the dataset dimensionality further. The downscaling transform consists of computing neighbor pixels' average luminosity, similar to the average pooling described in Section 2.2.2. By employing a downscaling factor of $f_{ds} = 5$, they reduced the number of features by $f_{ds}^2 = 25$ times. Despite this dimensionality reduction, no meaningful information was lost, as evidenced by a retained variance and cross-entropy analysis. A similar analysis was performed in this work, as described in Section 4.2.2.4. Images were also cropped to the natural Region of Interest (ROI) of the experiment, removing irrelevant objects such as the electrodes and the boiling chamber. A final dimensionality reduction step was transforming image representation using Principal Component Analysis (PCA) (MURPHY, 2012). A structural similarity analysis also showed that images acquired at 30 fps are uncorrelated, which helps in preventing biased estimates. In the task of classifying pool boiling images between natural convection, nucleate boiling, and film boiling (described in detail in Section 2.1), both SVMs and MLPs achieved over 90% accuracy in indirect visualization (described in Section 4.2.2.6), and approximately 99% of accuracy, precision, and recall in direct visualization (also explained in Section 4.2.2.6). An additional investigation also demonstrated that MLPs were more than 100 times quicker than SVMs in performing predictions.

In a subsequent study (HOBOLD; DA SILVA, 2018a), the authors showed that CNNs perform significantly better than SVMs and shallow MLPs in classifying pool boiling regimes, even without PCA. Based on this result, their third work (HOBOLD; DA SILVA, 2019b) evaluated the performance of CNNs in the visualization-based quantification of heat flux in pool boiling and corroborated the hypothesis that they are superior to MLPs in performance and in inference time. Like in Hobold and da Silva (2018b), the authors considered the differences between direct and indirect visualization. For heat fluxes greater than 10 W/cm^2 , where the boiling regime is either nucleate or film boiling, they obtained the error metrics reproduced in Table 1. Those results are used in this work in Section 4.4 to validate the training pipeline.

To achieve the results in Table 1, Hobold and da Silva (2019b) employed the CNN illustrated in Figure 10. The model architecture consists of: the input layer; a convolutional layer with 32 filters of 5×5 kernel size; a 2×2 max-pooling layer; a 200-unit dense layer; and a single-unit output head. A ReLU activation activates the convolutional and hidden dense layers. In contrast, the output unit is linearly activated to allow correct real-valued predictions. Overfitting is prevented by using a 50% dropout. Images were preprocessed using the same pipeline as Hobold and da Silva (2018b), including grayscaling, downscaling, and cropping to the ROI. In addition, they assessed the effect of the width of the visualization window. They discovered that images could be laterally cropped to 60% of their original size without significantly losing performance. As shown in the results in Table 1, models trained with indirect visualization perform significantly worse than

Table 1 – Reference metrics from Hobold and da Silva (2019b). Obtained for nucleate and film boiling regimes defined by $q'' \geq 10 \text{ W/cm}^2$. Validation and test metrics are presented as functions of the type of visualization: direct, in which the heater surface is visible in the images, or indirect, in which it is cropped out. Confidence intervals of 95 % around the mean were computed by bootstrapping using 1000 batches of 100 samples. Reproduced from Hobold and da Silva (2019b).

Metric	Unit	Direct		Indirect	
		Validation	Test	Validation	Test
R^2	—	$0.9828^{+0.0017}_{-0.0019}$	$0.9826^{+0.0018}_{-0.0020}$	$0.9557^{+0.0046}_{-0.0050}$	$0.9564^{+0.0049}_{-0.0050}$
MAPE	%	$7.37^{+0.46}_{-0.42}$	$7.62^{+0.44}_{-0.42}$	$10.60^{+0.56}_{-0.56}$	$10.35^{+0.52}_{-0.51}$
MAE	W/cm^2	$2.77^{+0.14}_{-0.14}$	$2.66^{+5.78}_{-2.55}$	$3.98^{+9.93}_{-3.85}$	$3.97^{+9.82}_{-3.83}$
MSE	$(\text{W/cm}^2)^2$	$13.03^{+1.37}_{-1.19}$	$13.19^{+1.49}_{-1.31}$	$33.55^{+4.05}_{-3.73}$	$32.87^{+3.92}_{-3.65}$

their direct-visualization counterparts, almost triplicating the MSE metric. As pointed out by Hobold and da Silva (2018b, 2019b), this is likely a consequence of the additional information that the wire and its neighborhood contain regarding bubble formation and departure, which is accessible to direct-visualization models only.

Finally, Hobold and da Silva (2019b) loaded their trained model onto a Raspberry Pi 3 Model B machine to investigate the applicability of their methodology to achieve real-time, low-cost heat flux measurement. Because of the availability of only 1 GB of RAM in that machine, they reduced their model size by cutting the number of filters in the convolutional layer to 16. By doing this, they showed that the Pi could perform approximately 7 predictions per second, which is a promising result. However, the RAM limitation demonstrates that further work is necessary to reduce the memory footprint of trained models.

Additionally, Hobold and da Silva (2019a) observed that even the classification accuracies of 99 % obtained in Hobold and da Silva (2018a) can be prohibitively low for the detection of the DNB to prevent the boiling crisis in thermal systems since it misclassifies one frame per 10 000 frames, or one frame each 333 s at 30 fps. Therefore, the authors introduced a method based on Bayesian inference on top of the previously trained CNN. This way, each image is classified depending on the CNN prediction for them and the classification of the previous frames. With this approach, it was demonstrated that DNB could potentially be detected with less than 10 frames.

The promising results obtained by Hobold and da Silva (2018b,a, 2019b,a) support the application of visualization-based machine learning models to pool boiling problems. Despite this, the authors pointed out that there is still room for significant improvements to their methodology. In particular, only three CNN were considered in Hobold and da Silva

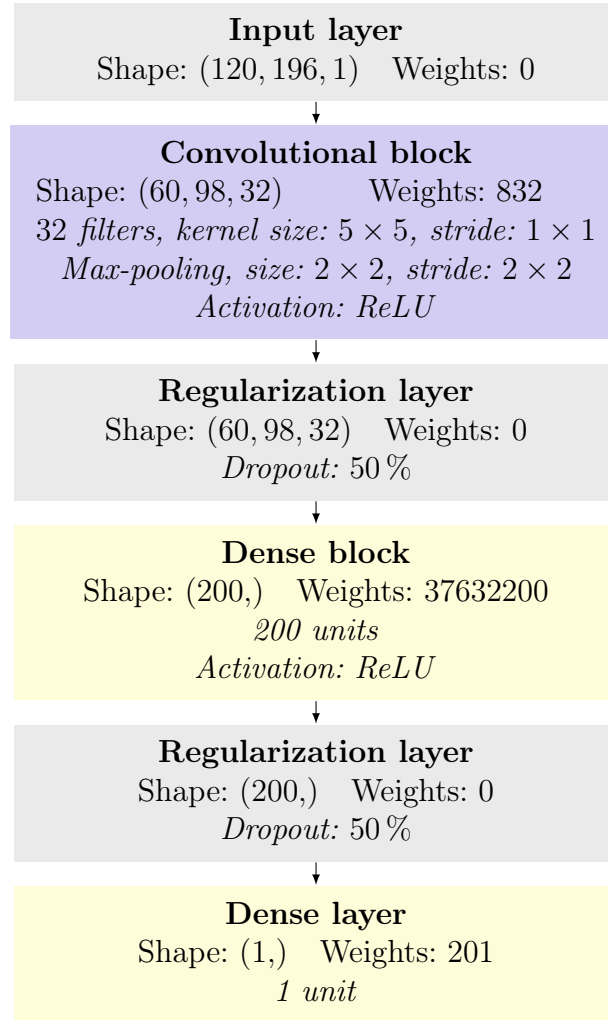


Figure 10 – Representation of the CNN utilized by Hobold and da Silva (2019b). A convolutional block is a sequence of three layers: a convolutional layer, an activation layer, and a max-pooling layer. Similarly, a dense block is a dense layer followed by an activation layer. Each layer is displayed alongside its outputs' shapes and the number of trainable weights it contributes to the model. The weights of dense layers are arranged in one-dimensional vectors whose shape is denoted as $(\cdot,)$, following the Python notation for tuples of unitary length.

(2019b), and a systematic search for an optimal architecture is still open for exploration. In addition, all studies were carried out in the same operating conditions with identical test samples. Consequently, the proposed methodology's generalization capability remains an open question. Moreover, the limited availability of RAM in low-cost hardware described by Hobold and da Silva (2019b) poses the need for smaller, more efficient models.

Scariot (2019) extended those studies by considering the quantification of the dissipated heat in pool boiling from a flat copper surface. This represents a change in the heater surface material and its geometry, which is likely to cause drastic changes in the system's behavior, as explained in Section 2.1.2. A single CNN architecture was designed containing six convolutional layers interspersed with three max-pooling layers and topped by a 256-unit hidden dense layer, followed by a single-unit output head. Figure 11 depicts

this architecture. Despite the bigger model employed, a worse performance than Hobold and da Silva (2019b) was obtained, with reported errors of up to 24% over the validation set and 33% over the test set. Two studies were carried out, and the reported performance metrics can be found in Table 2.

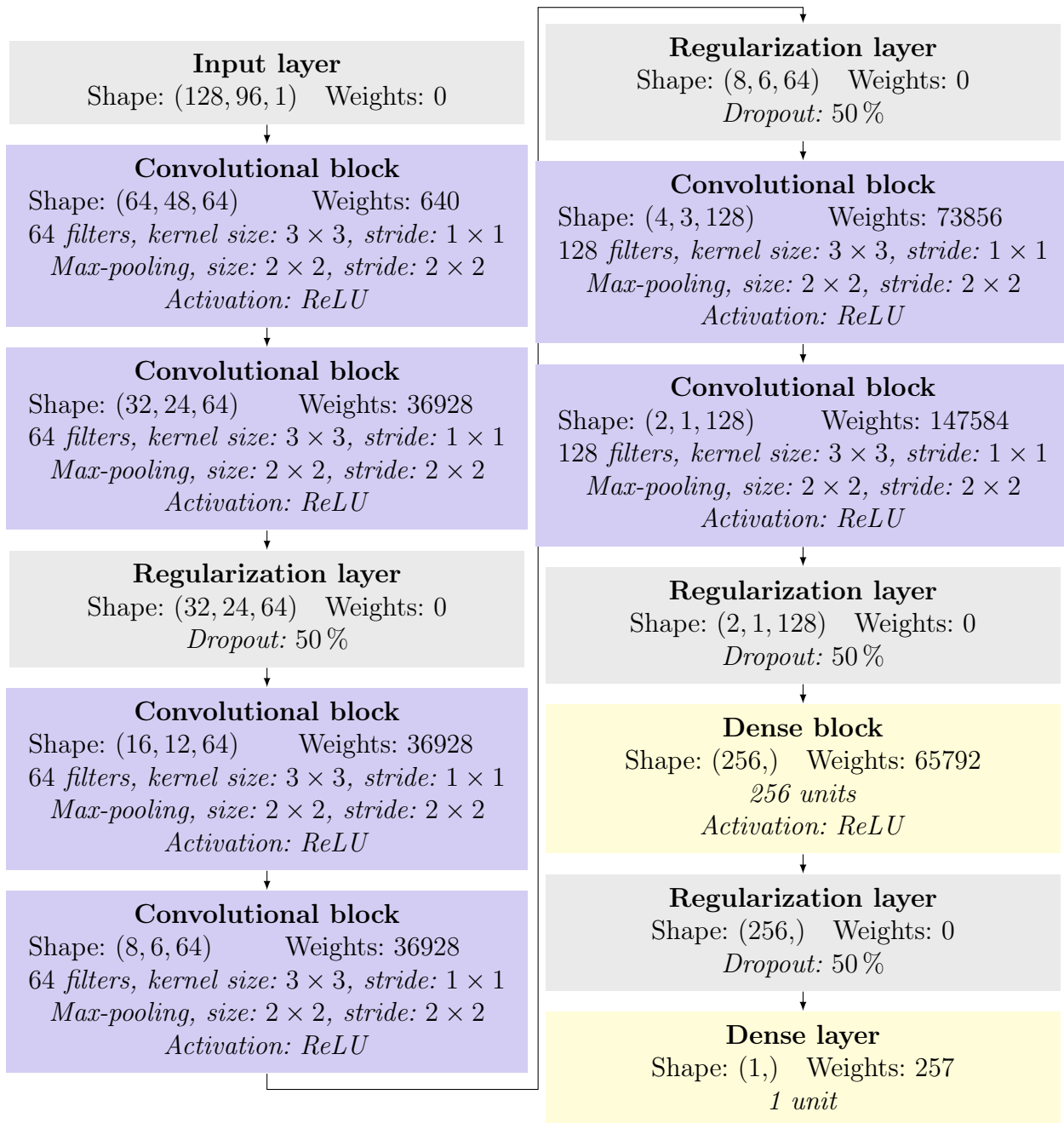


Figure 11 – Representation of the CNN utilized by Scariot (2019). The same notes for Figure 10 apply.

Scariot (2019) presented several reasons for this increase in the error metrics compared to the on-wire pool boiling results obtained by Hobold and da Silva (2019b): (a) on-wire boiling is intrinsically a bi-dimensional problem from the camera Point of

Table 2 – Reference metrics from Scariot (2019). Obtained for two different datasets generated by choosing different heat flux levels from the global dataset. Validation and test metrics are computed by bootstrapping 10 000 batches of 1000 samples each. Reproduced from Scariot (2019).

Metric	Unit	Study 1		Study 2	
		Validation	Test	Validation	Test
MSE	$(\text{W}/\text{cm}^2)^2$	47.0 ± 8.3	213 ± 34	45.0 ± 8.3	179 ± 26
MAE	W/cm^2	5.04 ± 0.44	10.88 ± 0.93	4.87 ± 0.43	10.00 ± 0.84
R^2	—	0.965 ± 0.006	0.865 ± 0.024	0.967 ± 0.007	0.839 ± 0.032

View (PoV), whereas boiling from a flat surface contains information (such as bubbles) in the depth direction, which is not captured; (b) the on-wire dataset also contained more heat flux steps, which may have allowed models to generalize better; (c) Hobold and da Silva (2019b) tested three different architectures, whereas Scariot (2019) considered only one—even though three sets of hyperparameters do not compose a significant search space, they may have provided sufficient performance gains; (d) in the studies carried out by Scariot (2019), the test set was generated by holding out some heat flux levels so that they were absent from the training set—this differs from the subsets used by Hobold and da Silva (2019b), which are sampled from all heat flux levels.

No studies were found in the existing literature applying AutoML techniques to find an optimal model architecture for phase-change problems. To the best of the author’s knowledge, this is the first work to attempt that.

2.4 PROBLEM SPECIFICATION

This Chapter introduced in Section 2.1 pool boiling as an important heat transfer phenomenon with broad applications. However, the prediction and modeling of pool boiling systems is still an active area of research. In particular, the effects of heater surface variability on pool boiling are challenging to consider. Additionally, Section 2.2 presented machine learning as a promising tool for computer vision problems, and Section 2.3 reviewed recent studies exploring the application of machine learning models to quantify the pool boiling phenomenon.

However, as indicated in Section 2.3, those studies did not consider the effects of heater surface variability on the pool boiling phenomenon. This is a significant gap in the literature, as the heater surface is a key component of the pool boiling system and considerably changes the system’s behavior. Moreover, searching for optimal machine learning architectures and training hyperparameters was not a goal of those studies, which leaves the door open for a systematic exploration of different models aiming at minimizing the generalization error.

As a result, this work aims to fill the gap in the literature by investigating the

performance of machine learning models when trained and evaluated on pool boiling datasets generated with different heater surfaces. In order to achieve this goal, this work investigates the effects of data preprocessing steps and the application of AutoML to reduce the generalization error of trained models. Figure 12 depicts the areas of study where this work intersects.

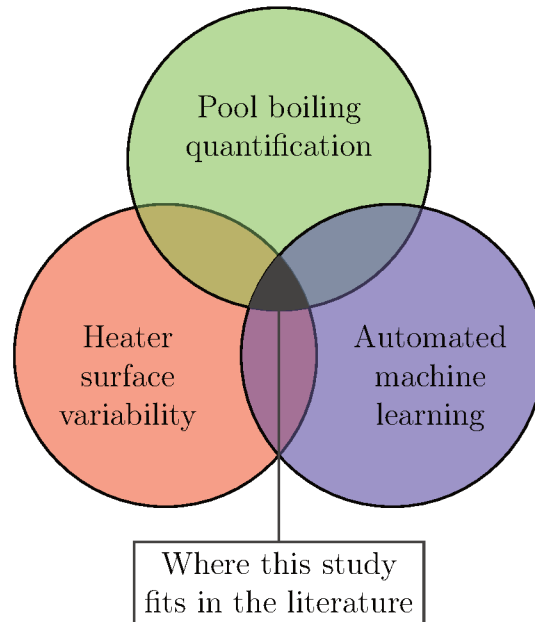


Figure 12 – The areas of study where this work intersects.

3 EXPERIMENTAL METHODOLOGY

In order to train machine learning models, it is necessary to collect data. This Chapter describes the pool boiling experimental setup employed to generate data for this work and the resulting datasets and their characteristics. Sections 3.1 and 3.2 describe the experimental apparatus employed in this work and the adaptations necessary to make that possible; Section 3.3 delineates the step-by-step procedure followed to gather experimental data; Section 3.5 estimates the uncertainties around the heat flux measurements; and Section 3.4 describes the datasets obtained by the experimental procedure.

3.1 EXPERIMENTAL APPARATUS

The experimental apparatus utilized in this work to obtain pool boiling data comprises a boiling chamber, a camera, and a Data Acquisition System (DAQ)^[1]. The boiling chamber is a borosilicate-glass cylinder with an inner diameter of 144 mm, a height of 200 mm, and a wall thickness of 5 mm enclosed with stainless steel plates at the top and bottom. The top plate contains openings to the atmosphere to help keep the inner pressure of the chamber approximately constant.

Inside the chamber, two copper electrodes separated by 60 mm sustain the test sample. A 1500 W Direct Current (DC) power supply provides electric current to the test sample by imposing electric tension between the electrodes. Voltage and current are manually adjusted. The boiling chamber is filled with deionized water to avoid current leakage and ensure consistent behavior between experimental runs and the results in the literature. Figure 13 illustrates the experimental apparatus, including a photograph and two schematic representations. Examples of the frames obtained with this setup are shown in Section 3.4.

The electric potential difference V_{sample} between the test sample's terminals is measured at the base of the electrodes, whereas the electric current I that flows through the test section is calculated by measuring the voltage drop V_{shunt} at an ammeter shunt installed in series with the circuit. The ammeter shunt has a resistance of $R_{\text{shunt}} = 4 \text{ m}\Omega$ known with $\pm 0.5\%$ of uncertainty. The current I is calculated according to Ohm's Law, $V_{\text{shunt}} = R_{\text{shunt}}I$. The thermal power q dissipated by the test sample by the Joule effect is then calculated as $q = V_{\text{sample}}I$. Finally, the heat flux q'' can be calculated as $q'' = q/A_s$ according to Equation (2). Figure 14 contains an electrical diagram for the system, demonstrating the two sections where electric tension is measured.

A DAQ acquires all electric data during the experiments. The electric tension between the test sample terminals is measured with an uncertainty of $\pm 6230 \mu\text{V}$ and the

^[1] In this work, the term DAQ is used to refer to both *data acquisition* in general and *data acquisition systems*. This confusion is an intentional simplification commonly used in the literature and in the industry.

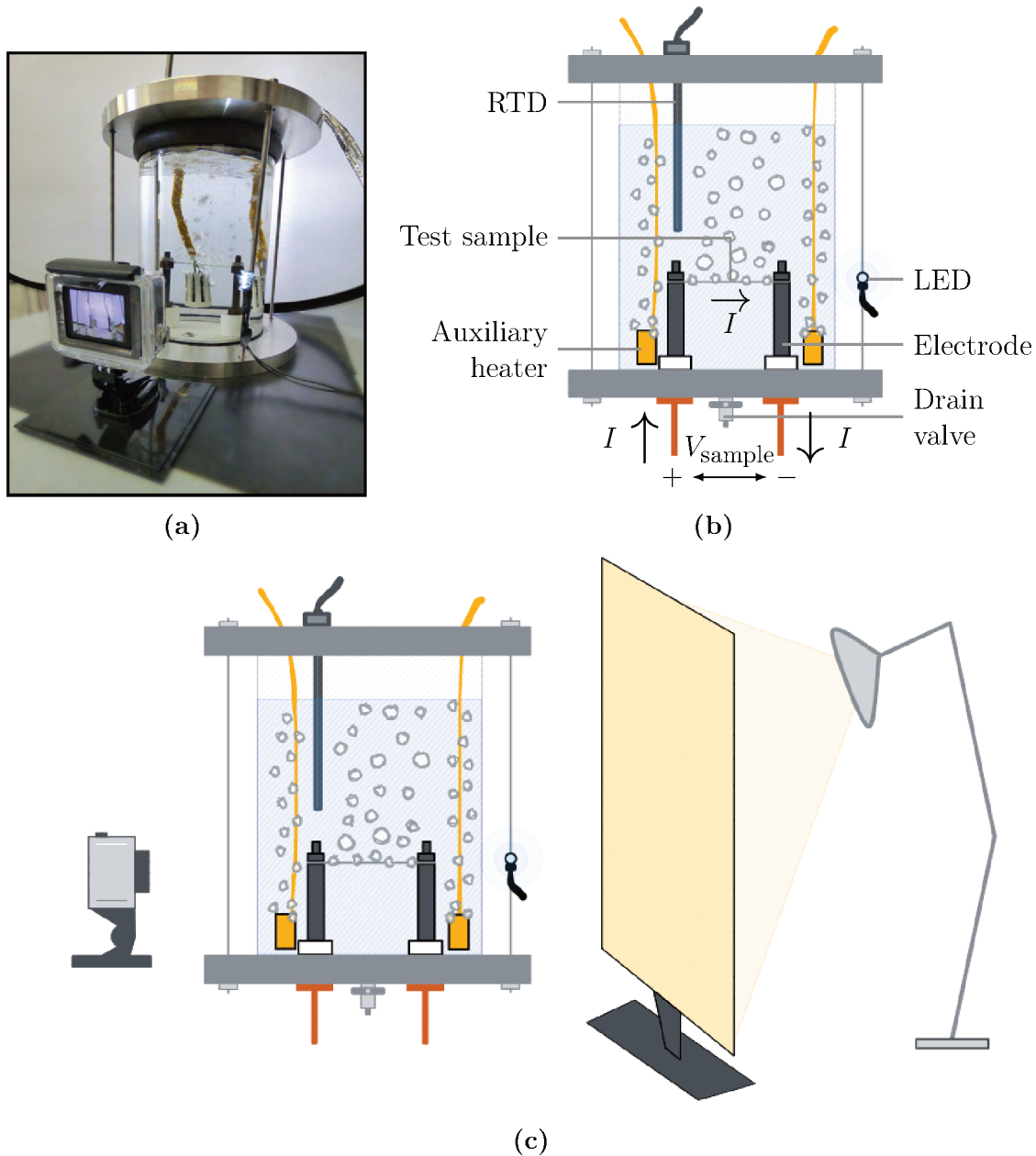


Figure 13 – Pool boiling setup, including (a) a photograph demonstrating the positioning and scale of the components of the experimental apparatus; (b) a schematic representation of the boiling chamber showing the test sample connected to the electrodes, the path of the electric current I and the auxiliary subsystems; and (c) an illustration of the visualization equipment displaying the relative positioning of the camera, the boiling chamber, the light diffuser, and the backlighting lamp. Note that (b) and (c) are not to scale.

voltage drop at the shunt is acquired with $\pm 174 \mu\text{V}$ of uncertainty.

Two 750 W heaters maintain the bulk liquid temperature T_{∞} close to the saturation temperature T_{sat} . The heaters are controlled manually via a dimmer. A RTD transducer is installed in an eccentric position distantly from the test sample and the heaters to measure the bulk liquid temperature. Because of its eccentric position, the RTD measures

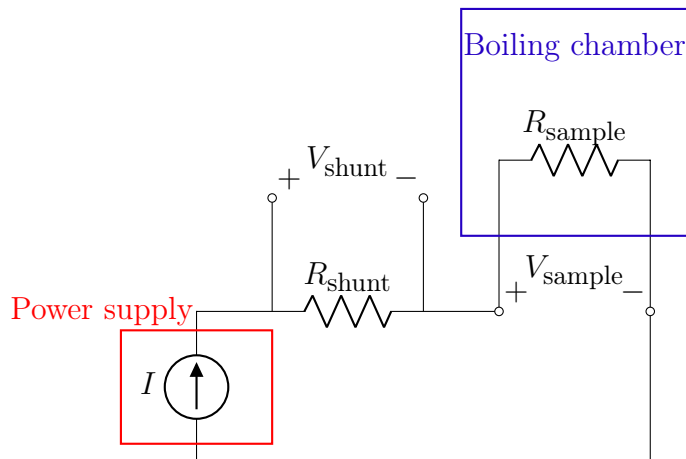


Figure 14 – Electrical diagram for the pool boiling experimental apparatus.

the descending liquid temperature, which is a better approximation for the bulk liquid temperature than the rising, overheated vapor bubbles. Changes in the liquid temperature translate into changes in the RTD’s internal resistance, which is, in turn, measured by the DAQ using the 4-wires method with a maximum uncertainty of ± 0.35 °C. The RTD was calibrated to a third-order polynomial using an ultrathermostatic water bath and a high-precision reference thermometer.

A conventional Digital Single-Lens Reflex (DSLR) HERO4 (GoPro) camera positioned in front of the chamber acquires images in 2.7 K quality at a rate of 30 fps (frames per second) with a linear Field of View (FoV), that is, without distortion. In order to prevent the machine learning algorithms from using the scene background (irrelevant to the boiling process) to learn and make predictions, a photographic diffuser, backlit with a Light-Emitting Diode (LED), is installed. With that, the image background remains unchanged from one experimental run to another.

Because of the cylindrical shape of the boiling chamber, video frames can be slightly distorted, especially near the borders of the ROI. Despite this, no correction was made as a deliberate simplification of the preprocessing pipeline described in Section 4.2.2. If any correction is necessary, trained machine learning models are expected to learn them.

3.2 ADAPTATION OF THE EXPERIMENTAL APPARATUS

As part of this work, the experimental apparatus left by Scariot (2019) was reverted to the format employed by Hobold and da Silva (2018a,b, 2019a,b) to collect pool boiling data with different one-dimensional test samples instead of bi-dimensional surfaces. This Section describes the changes and fixes applied.

The main adaptation consisted in replacing the copper block at the bottom of the boiling chamber with the Polytetrafluoroethylene (PTFE) base sustaining the two copper electrodes built by Hobold and da Silva (2018a). After this substitution, several additional modifications and fixes were implemented to ensure the correct behavior of the apparatus:

- **maintenance and repair:** (a) repair of the electric circuitry, including the replacement of defective wires and cables; (b) corrective maintenance of the fastening and sealing elements, like the substitution of an O-ring that lost elasticity with age and reinforcement of the high-temperature silicone sealant that showed signs of wearing and liquid leakage;
- **substitution of equipment:** (a) replacement of the previous DAQ with the one described in Section 3.1, including the voltage, current, and temperature input modules; (b) substitution of the ammeter shunt by a higher-precision model, as described in Section 3.1; (c) replacement of the high-pressure outlet valve utilized by Hobold and da Silva (2019b) with a gate valve drain; (d) substitution of the previous camera model by the HERO4 (GoPro) model described in Section 3.1;
- **addition of auxiliary components:** (a) construction of a tension divider to measure the voltage at the base of the copper electrodes. This divider is necessary due to a limit imposed by the DAQ voltage input module of at most 10 V per terminal. Since the maximum tension delivered by the power supply is 30 V, three identical resistors were assembled in series, and the potential drop across each resistor was read by different terminals; (b) installation of a manually-adjustable dimmer to control the thermal power dissipated by each auxiliary heater in the range 0–750 W. This is important to avoid extreme changes in the bulk liquid temperature which could cause the borosilicate glass to break; (c) implementation of a signaling subsystem based on a LED. In this subsystem, a LED is positioned inside the camera’s FoV, outside the boiling ROI. When the LED is activated, an electric signal is simultaneously sent to the DAQ, allowing for the synchronization between the recorded video and the voltage, current and temperature data. This signaling subsystem is also manually activated. (d) electrostatic shielding of all signal wires and cables to reduce the electromagnetic interference in the measurements caused by near machinery (for instance, the power supply). Both electrostatic shields and the boiling chamber metallic skeleton are grounded.

Those adaptations took a considerable time, mainly because some did not seem necessary at first, like the electrostatic shielding – their need only became evident as issues appeared during the experiments. Other changes, such as adding the dimmer to limit the power dissipated by the auxiliary heaters, were recommended by the previous authors based on their own experience, even though this is not explicitly reported in their work (SCARIOT, 2019).

In addition to the changes in the experimental setup itself, due to the lack of available licenses when the experiments were run, a custom, minimal DAQ software was implemented in Python 3.7 to read, store, and display measured data in real time. This program

was built on top of Nidaqmx (NATIONAL INSTRUMENTS, 2017), PyQtGraph (CAMPAGNOLA, 2019), Modin (MODIN PROJECT, 2018), Matplotlib (HUNTER, 2007), Numpy (HARRIS et al., 2020) and SciPy (VIRTANEN et al., 2020).

Despite the best efforts made to replicate the experiments from Hobold and da Silva (2018b,a, 2019b,a), some differences between the data gathered, and hence the trained models, are expected. Many factors can explain possible deviations between the experimental results, such as (a) the utilization of a different camera model, which may change the characteristics of the captured frames; (b) the employment of different DAQ components, which may change the characteristics of the measured data; (c) natural differences in the lighting and positioning of the various components of the experimental apparatus; and (d) the use of different test samples, as described in Section 3.4. However, the possible differences in the training data have not affected the trained models' final performance, as shown in Section 4.4.

Additional changes can be recommended for future studies to simplify the experimental procedure, detailed in Section 3.3, or allow further analyses. Section 6.2 presents those ideas and other suggestions for future work.

3.3 EXPERIMENTAL PROCEDURE

The first stage in each experimental run consists of (a) replacing the test sample, if applicable to that run; (b) filling the boiling chamber with deionized water; (c) adjusting the camera, ensuring that it is centered and correctly placed; (d) executing the DAQ program in order to monitor the subsequent steps; (e) turning the auxiliary heaters on in order to bring the liquid temperature T_∞ to the saturation temperature T_{sat} ; and (f) waiting for approximately 1 min for degassing after T_{sat} is reached. The degassing step and the use of deionized water are essential to ensure that the liquid is pure water since mixtures tend to have different properties compared to pure liquid. After degassing is done, the data-gathering stage begins.

Each dataset was generated starting from thermal power of $q = 0\text{ W}$, in the natural convection regime, and increasing in steps of 5 W until sample failure per burnout, as illustrated in Figure 15a. Each power level required fine adjustment of the imposed current and voltage until stabilization. After the correct nominal power was achieved, LED synchronization was executed by turning the LED on and off, emitting a signal to the DAQ and allowing posterior synchronization between the recorded video frames and the DAQ measurements. Subsequently, an 1 min video was recorded, capturing the boiling phenomenon. After the recording was complete, the thermal power was increased to the next level, and the data-gathering stage restarted. This process is illustrated in Figure 15b.

Note that this experimental procedure, together with the boiling setup as described in Sections 3.1 and 3.2, is sufficient to keep all important system parameters listed in Section 2.1.2 fixed apart from the heater surface:

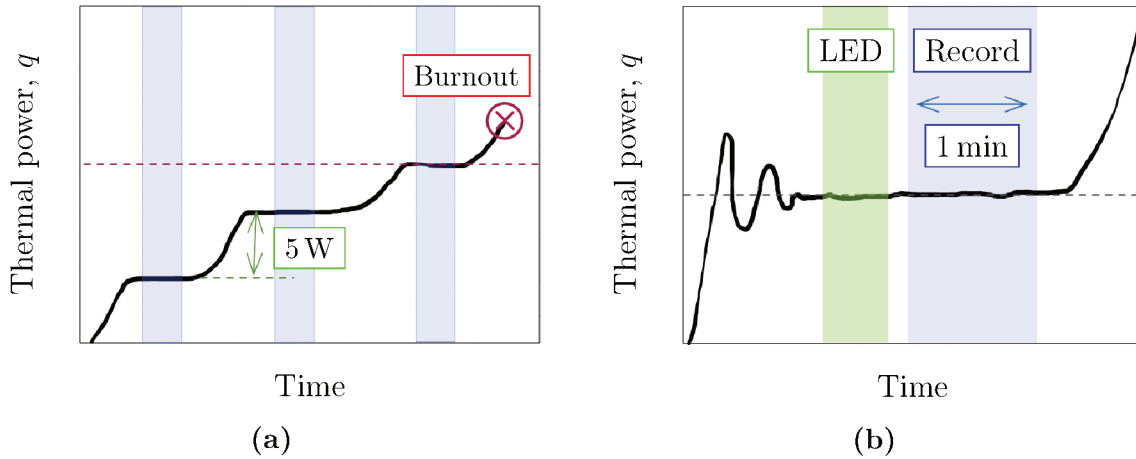


Figure 15 – Illustration of the experimental procedure, including (a) multiple thermal power steps increasing in steps of 5 W/cm² until burnout, and (b) the sequence of thermal power stabilization, LED synchronization, and video recording. Note that this illustration is conceptual and does not contain real experimental data.

- **pressure:** the system pressure is kept constant by leaving the chamber open to the atmosphere and constantly feeding water into the chamber to keep the liquid level stable as the water evaporates;
- **subcooling:** subcooling is prevented by ensuring that the bulk liquid is always at its saturation temperature, as measured by the RTD;
- **gravity:** all experiments were run at the same location;
- **mode of the test:** the experimental runs consistently increased the dissipated thermal power in steps of 5 W, and each step was kept constant for 1 min after stabilization of the measured heat flux;

3.4 EXPERIMENTAL CASES

Based on the procedure described in Section 3.3, four pool boiling experimental cases were obtained. This Section describes them and their characteristics.

Four test samples were utilized in the experiments:

- a sample from NI80-020-200 (Omega Engineering) nichrome wire with a diameter of $(510 \pm 4) \mu\text{m}$, slightly different from the value of $(451.7 \pm 0.4) \mu\text{m}$ reported by Hobold and da Silva (2019b);
- a sample from NI80-010-200 (Omega Engineering) nichrome wire with a diameter of $(250 \pm 4) \mu\text{m}$; and
- two samples from NCRR-17-100 (Omega Engineering) nichrome ribbon $(79 \pm 4) \mu\text{m}$ thick and $(1590 \pm 4) \mu\text{m}$ wide.

The test samples were cleaned before being connected to the electrodes using deionized water. Apart from this, no special surface treatment or texturing was applied to the surfaces. Hence, their characteristics are the same as those delivered by the supplier Omega Engineering. Appendix A reproduces the datasheets for all samples.

For each sample, an experimental dataset was obtained according to the experimental procedure in Section 3.3:

- **Large Wire (\mathcal{D}^{LW}):** dataset generated from the NI80-020-200 (Omega Engineering) nichrome wire. Since this wire has the largest diameter, the resulting dataset is referred to as the *large wire dataset* and is denoted by \mathcal{D}^{LW} . In addition, because this dataset is the most similar to the one utilized by Hobold and da Silva (2019b), this dataset is also referred to as the *baseline dataset*. Figure 16 shows an example of a video frame acquired for this dataset at the nominal power of 75 W;
- **Small Wire (\mathcal{D}^{SW}):** dataset generated from the NI80-010-200 (Omega Engineering) nichrome wire. Since this wire is the one with the smallest diameter, the resulting dataset is referred to as the *small wire dataset* and is denoted by \mathcal{D}^{SW} ;
- **Horizontal Ribbon (\mathcal{D}^{HR}):** dataset generated from one of the NCCR-17-100 (Omega Engineering) nichrome ribbon samples connected to the electrodes in the horizontal position. Consequently, the resulting dataset is referred to as the *horizontal ribbon dataset* and is denoted by \mathcal{D}^{HR} .
- **Vertical Ribbon (\mathcal{D}^{VR}):** dataset generated from the other NCCR-17-100 (Omega Engineering) nichrome ribbon sample. In order to produce this dataset, the ribbon was connected to the electrodes in the vertical position. For this reason, the resulting dataset is referred to as the *vertical ribbon dataset* and denoted by \mathcal{D}^{VR} ;

The frame shown in Figure 16 is a *raw* frame reproduced exactly as the camera captures it. It contains elements from the experimental environment, including components of the experimental setup, such as the LED and the light diffuser. However, this is unsuitable for training machine learning models, as described in Chapter 4. Hence, Section 4.2.2.2 demonstrates the process of cropping frames to the ROI.

Table 3 presents representative frames from each dataset in the natural convection, partial nucleate boiling, and fully developed nucleate boiling regimes. The frames were cropped to the ROI to illustrate the data fed to the machine learning models. Note from Table 3 that not all frames have the same size because the ROI changed between experimental runs. One of the causes for this variability is that, at high heat fluxes, the auxiliary heaters moved considerably inside the chamber due to the intensified movement of the liquid, and care had to be taken to ensure that they did not appear in any of the frames after cropping.



Figure 16 – Example video frame obtained at 75 W for the baseline, large wire dataset \mathcal{D}^{LW} . The video frame contains environmental clues such as the LED or reflexes, which are cropped out of the ROI, as described in Section 4.2.2.

Regarding the boiling parameters, due to the 5 W steps used to increase the dissipated heat flux, events such as the ONB or the DNB can only be observed within a resolution of 5 W. For instance, the last stable power level achieved for \mathcal{D}^{LW} before burnout was 85 W (equivalent to the heat flux level of 86.80 W/cm^2). Hence, it can be inferred that the CHF was within the range of 85–90 W (or $86.80\text{--}92.20 \text{ W/cm}^2$). A similar reasoning applies to finding the ONB in an experimental run. Table 4 presents the ranges for the ONB and the DNB observed in each experimental case. The results demonstrate good agreement with the parametric effects described in Section 2.1.2. In particular, the DNB happened earlier on the vertical ribbon, corroborating the statement that inclined surfaces present a lower CHF.

Table 3 – Representative frames from each dataset in the natural convection, partial nucleate boiling, and fully developed nucleate boiling regimes. Note that not all frames have the same size since the ROI changed between experimental runs.

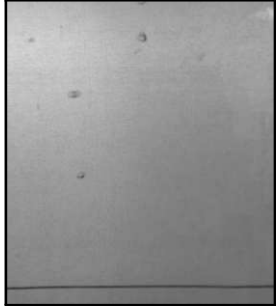
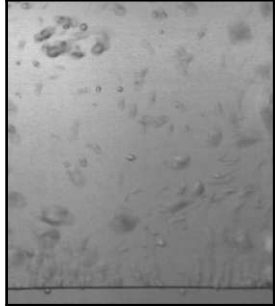
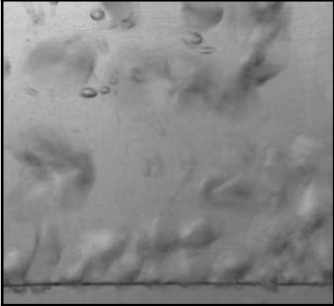


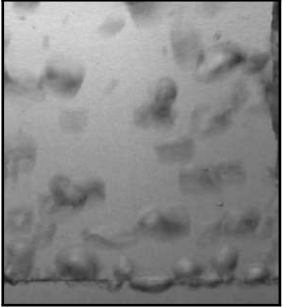
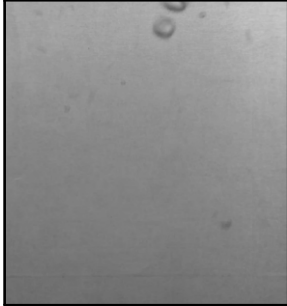
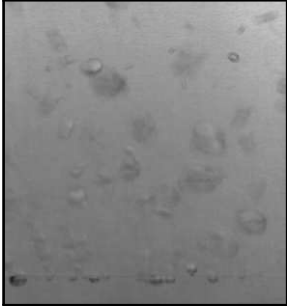
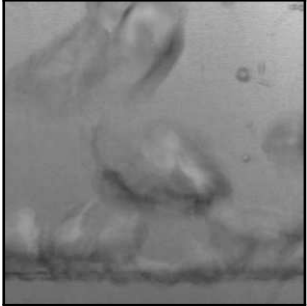
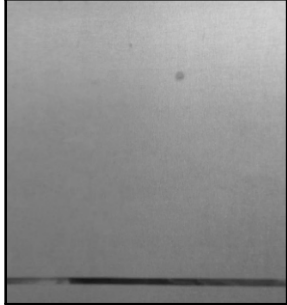
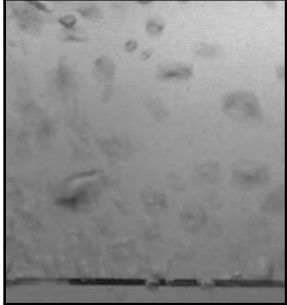
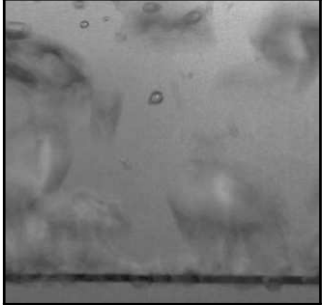
Dataset	Natural convection	Partial nucleate boiling	Fully developed nucleate boiling
Large wire			
Small wire			
Horizontal ribbon			
Vertical ribbon			

Table 4 – Approximate ranges for the thermal power and heat flux at the ONB and the DNB for each experimental dataset. The main ranges are the nominal thermal power levels surrounding the event, whereas the values in parentheses are the average heat flux calculated for those levels. Because of natural variability in the measured heat flux, the averages in parentheses do not necessarily satisfy Equation (2). In addition, the higher ends of the DNB intervals are unknown since no experimental data is available for those points.

Dataset	ONB	DNB
Large wire	10–15 W (9.88–14.38 W/cm ²)	85–90 W (86.10–? W/cm ²)
Small wire	5–10 W (10.36–20.76 W/cm ²)	40–45 W (81.03–? W/cm ²)
Horizontal ribbon	5–10 W (2.40–4.55 W/cm ²)	185–190 W (85.76–? W/cm ²)
Vertical ribbon	10–15 W (4.78–6.75 W/cm ²)	170–175 W (78.60–? W/cm ²)

3.5 UNCERTAINTY ANALYSIS

The uncertainty analysis is a crucial step in any experimental work since it provides a quantitative measure of the experimental error. In this work, the uncertainty analysis is performed as explained in Appendix B.

The heat flux q'' can be calculated from measurable or known quantities according to the equations presented in Section 3.1:

$$q'' = \frac{q}{A_s}, \quad q = V_{\text{sample}}I, \quad I = \frac{V_{\text{shunt}}}{R_{\text{shunt}}}. \quad (16)$$

This chain of equations can be utilized to calculate the standard uncertainty of the heat flux q'' according to Equation (25). It can be shown that

$$\frac{\hat{U}(q'')}{q''} = \sqrt{\left(\frac{\hat{U}(V_{\text{sample}})}{V_{\text{sample}}}\right)^2 + \left(\frac{\hat{U}(V_{\text{shunt}})}{V_{\text{shunt}}}\right)^2 + \left(\frac{\hat{U}(R_{\text{shunt}})}{R_{\text{shunt}}}\right)^2 + \left(\frac{\hat{U}(A_s)}{A_s}\right)^2}. \quad (17)$$

In this equation, $\hat{U}(V_{\text{sample}}) = \pm 6230 \mu\text{V}$ and $\hat{U}(V_{\text{shunt}}) = 174 \mu\text{V}$ are constant type B uncertainties given in Section 3.1. The ratio $\hat{U}(R_{\text{shunt}})/R_{\text{shunt}} = \pm 0.5\%$ is also a constant given in Section 3.1. Additionally, the ratio $\hat{U}(A_s)/A_s$ is constant in each dataset, depending only on the test sample geometry and its measurement uncertainty. Consequently, during the experiments, the heat flux uncertainty changes depending only on the heat flux and the voltage drop on the test sample and the shunt resistor.

Additionally, it can be inferred from Equation (17) that the heat flux uncertainty increases with the heat flux. Therefore, it has a minimum close to zero and a maximum at the maximum heat flux. Table 5 presents the maximum heat flux uncertainties obtained for each experimental dataset.

Table 5 – Maximum heat flux uncertainties for each experimental dataset. The standard and expanded uncertainty, obtained with a coverage factor of 2, are presented. For convenience, the simplified uncertainty, as it would appear in a measurement report, is also shown.

Dataset	Uncertainty		
	Standard	Expanded	Simplified
Large wire	$\pm 0.633 \text{ W/cm}^2$	$\pm 1.266 \text{ W/cm}^2$	$\pm 1 \text{ W/cm}^2$
Small wire	$\pm 1.113 \text{ W/cm}^2$	$\pm 2.227 \text{ W/cm}^2$	$\pm 2 \text{ W/cm}^2$
Horizontal ribbon	$\pm 0.522 \text{ W/cm}^2$	$\pm 1.044 \text{ W/cm}^2$	$\pm 1 \text{ W/cm}^2$
Vertical ribbon	$\pm 0.483 \text{ W/cm}^2$	$\pm 0.966 \text{ W/cm}^2$	$\pm 1 \text{ W/cm}^2$

3.6 TEMPERATURE MEASUREMENT

A natural extension to the experimental apparatus is measuring the test sample surface temperature. This would make it possible to train machine learning models to

predict temperature values, which may be more valuable in industrial applications where limiting the system temperature is mandatory to ensure safe operation. Furthermore, if the test sample surface temperature is available in addition to heat flux measurements, it is possible to calculate the heat transfer coefficient and manipulate the system toward the optimal point of efficiency. In this work, several attempts were carried out to measure the test sample temperature. However, none of them could deliver results with low enough uncertainty to make that data useful.

Therefore, measuring the heater surface temperature is left as a suggestion for future studies. If that is done, models can be trained to estimate the heat transfer coefficient instead of the heat flux, which might be more interesting for specific applications.

4 MACHINE LEARNING METHODOLOGY

This Chapter explains the machine learning methodology employed in this work. Section 4.1 describes the implementation of the training pipeline and the execution environment. Section 4.2 describes the datasets and the preprocessing steps applied to them. Section 4.3 details the training pipeline that takes datasets as inputs and outputs trained machine learning models. Section 4.4 demonstrates the correctness of the preprocessing and training pipelines by comparing the results obtained with the ones from Hobold and da Silva (2019b). Finally, Section 4.5 describes the AutoML algorithm employed in the AutoML studies.

Regarding the development of machine learning systems, Goodfellow, Bengio, and Courville (2016) recommend a practical methodology of four steps:

- **Choosing performance metrics:** determining the most important metrics to optimize is essential for designing machine learning models. The performance metrics guide decision-making and the evolution of models during training and hyperparameter tuning. This Thesis utilizes the MSE as the primary metric for comparing models and optimizing the data preprocessing and training pipelines. The MSE is a suitable choice for regression tasks, as it measures the average squared difference between the predicted and the true targets. In addition, it is employed as the loss function in the training pipeline described in Section 4.3. Other metrics are also presented for information and easier comparison with the literature. However, they are not utilized to make decisions.
- **Defining baseline models:** a baseline model is the starting point on top of which analyses are performed. In this Thesis, the baseline model is a CNN of the same architecture as the best model found by Hobold and da Silva (2019b). It is trained with the same preprocessing pipeline and hyperparameters as in their work. The results from training this model are presented in Section 4.4 and are utilized to measure progress in the results from Chapter 5.
- **Gathering additional data:** in case the performance of the baselines is not satisfactory, Goodfellow, Bengio, and Courville (2016) recommends gathering additional data as the best option for improving it. However, as demonstrated in Section 4.4, this step is unnecessary for this work.
- **Selecting hyperparameters:** the last step consists of choosing the set of hyperparameters that specify the data preprocessing pipeline and the training algorithm. Hyperparameters can be selected either manually or via AutoML, explained in Section 2.2.4. In this Thesis, data preprocessing hyperparameters are chosen manually based on the results from Section 5.1. The training hyperparameters, including the optimizer and the model architecture, are selected via AutoML and are presented in Section 5.3.

4.1 IMPLEMENTATION AND EXECUTION

All computation was performed on an Ubuntu 20.04.5 LTS machine with 32 GB of 3200 MHz RAM, an 11th Gen Intel Core i7-11800H central processing unit running at 2.3 GHz, and an NVIDIA GeForce RTX 3070 Graphics Processing Unit (GPU) with the CUDA 11.5 (NICKOLLS et al., 2008) toolkit installed. Persistent data was stored on a 2 TB hard disk drive.

The entire training and evaluation pipeline was implemented in Python 3.10 (PYTHON CORE TEAM, 2015) and made available at Comelli (2023). The codebase was written following multiple paradigms with a strong focus on immutable data structures and type correctness. Lazy evaluation is employed to delay computation (for instance, data pre-processing) until it is needed, thus preventing the waste of computational resources and reducing memory consumption. In addition, persistent caching allows the re-use of previously calculated results across different pipeline runs. Many libraries were utilized either as direct or indirect dependencies in the various subsystems of the pipelines, the main ones being (COMELLI, 2023):

- NumPy 1.23.5 (HARRIS et al., 2020), scikit-image 0.19.3 (WALT et al., 2014), and TensorFlow 2.10.1 (ABADI et al., 2015d), for the image pre-processing and analysis described in Section 4.2.2;
- TensorFlow 2.10.1 (ABADI et al., 2015d), as the deep learning framework, accessed via the Keras API (CHOLLET et al., 2015). Preprocessed images are assembled in TensorFlow dataset pipelines and fed to models defined using Keras for training;
- KerasTuner 1.1.3 (O'MALLEY et al., 2019) and AutoKeras 1.0.20 (JIN, Haifeng; SONG; HU, 2019) for AutoML, described in Section 2.2.4.

4.2 DATASETS

The definition of the datasets is the first step in the training pipeline, as it determines the scope of the problem and the available data to train and evaluate the machine learning models. This Section describes the datasets utilized in this work and the preprocessing steps applied to them. All datasets were generated by manipulating the experimental data obtained as explained in Chapter 3.

4.2.1 Dataset splitting

The splitting between the training, validation, and test sets is the first decision to be made in order to ensure that the datasets are well-defined. In particular, to allow an unbiased estimation of model performance, the test set must not be employed to train models or optimize hyperparameters.

In this work, the dataset splitting is done using the hold-out algorithm, in which those subsets are disjoint and complementary to each other, and remain fixed during all runs of the training pipeline. This is in opposition to other algorithms, such as the k -fold cross-validation, in which the training and validation datasets are randomly sampled from the global dataset k times in order to train and evaluate a model (GOODFELLOW; BENGIO; COURVILLE, 2016). The hold-out splitting is considerably less computation-intensive than its alternatives since the splits are defined only once and re-used across all subsequent executions of the program. According to Goodfellow, Bengio, and Courville (2016), the hold-out splitting algorithm is usually valid when the global dataset comprises enough data for the validation and test metrics to be computed with confidence, as is the case in this study.

Goodfellow, Bengio, and Courville (2016) recommend assigning 80 % of the *training data* to the training set and the remaining 20 % to the validation set. In addition, it is advisable to keep the validation and test sets nearly the same size. For this reason, this work assigns 70 % of each heat flux level to the training set, 15 % to the validation set, and 15 % to the test set. Table 6 presents the resulting subset sizes for each dataset. Similarly to Hobold and da Silva (2019b), the splitting is done by randomly subsampling examples from each heat flux level separately, an important consideration to ensure a balanced representation of each level in the subsets. The splitting was done once and re-used across all executions of the training pipeline in order to avoid the accidental sharing of examples between those runs.

Table 6 – Dataset split sizes. Each subset (training, validation and test) is a disjoint subsample of the global dataset. The split sizes are shown along with the total dataset size for each experimental dataset.

Dataset	Training set (70 %)	Validation set (15 %)	Test set (15 %)	Total (100 %)
Large wire	26 416	5661	5662	37 739
Small wire	12 950	2773	2775	18 498
Horizontal ribbon	58 774	12 593	12 594	83 961
Vertical ribbon	51 276	10 987	10 987	73 250

The dataset splits are utilized as recommended by Goodfellow, Bengio, and Courville (2016) and Chollett (2018): (a) the training set is employed to train machine learning models and find the weights described in Section 2.2; (b) the validation set is employed to estimate the generalization error of trained models. With this, it is possible to tune hyperparameters, define early stopping criteria and compare different models; (c) finally, the test set is employed to estimate the unbiased generalization error of models. In order to reduce bias during this estimation, the test metrics cannot be utilized to optimize hyperparameters. For this reason, this work only uses the test metrics for reporting and communication, but no design decisions are made based on them.

The significantly reduced number of frames in the small wire dataset is due to the earlier DNB occurring naturally on smaller heater surfaces, as explained in Section 2.1.2 and verified in Table 4. Since all experimental runs consist of 1 min records obtained in steps of 5 W, the anticipated DNB results in fewer heat flux levels and, consequently, fewer samples. Similarly, due to the decrease in the CHF on inclined surfaces, the horizontal ribbon dataset contains more samples than the vertical ribbon dataset.

4.2.2 Image preprocessing

Data preprocessing is one of the critical stages in developing machine learning models since it determines the learning substrate for machine learning models to consume. Despite this, as Goodfellow, Bengio, and Courville (2016) suggested, computer vision problems require relatively little preprocessing. Image preprocessing steps are usually designed to remove irrelevant variability or information from the dataset.

Preprocessing steps that reduce the number of features in an example represent a form of *dimensionality reduction* (BISHOP, 2006) and usually cause the loss of information. This reduced amount of information available can either deteriorate or improve performance, depending on the removed pieces of information being relevant or containing excessive noise. Additionally, dimensionality reduction helps reduce the computational complexity of machine learning models, which is essential for developing real-time applications.

This Section describes the image preprocessing pipeline employed in this work to reduce data dimensionality and improve the performance of machine learning models. The resulting preprocessing pipeline transforms images as illustrated in Figure 17. Similar preprocessing steps were utilized by Hobold and da Silva (2018b, 2019b) and Scariot (2019), proving their effectiveness in the context of heat flux estimation.



(a) Original.

(b) Preprocessed.

Figure 17 – Demonstration of the results of the preprocessing pipeline, with (a) the original image and (b) its preprocessed version.

4.2.2.1 Image pixel format conversion

Whenever images of a given pixel data type are used in processing pipelines that expect a different data type, data may be accidentally corrupted by rounding or clipping. This fact was overlooked in the initial stages of this Thesis and proved to be a difficult issue to detect.

In order to prevent data corruption, all images were appropriately converted to 32-bit-precision floating-point numbers (NumPy’s `float32` data type) using TensorFlow’s (ABADI et al., 2015d) converters before being fed into the subsequent stages of the pipeline. This conversion does not change the dataset’s contents, only its representation. Floating-point number types are required in the preprocessing steps of grayscaling (Section 4.2.2.3), downscaling (Section 4.2.2.4), and standardization (Section 4.2.2.8).

4.2.2.2 Region of interest cropping

The pipeline’s first dimensionality reduction preprocessing step consists of cropping the experimental images to the correct ROI, as illustrated in Figure 18. Table 7 presents the image shapes before and after this preprocessing step. Note that the final image shapes are not necessarily equal across all datasets. This happened because the auxiliary heaters moved naturally due to the movement of the liquid inside the boiling chamber. Hence, different datasets required different cropping positions to ensure they are outside the ROI.

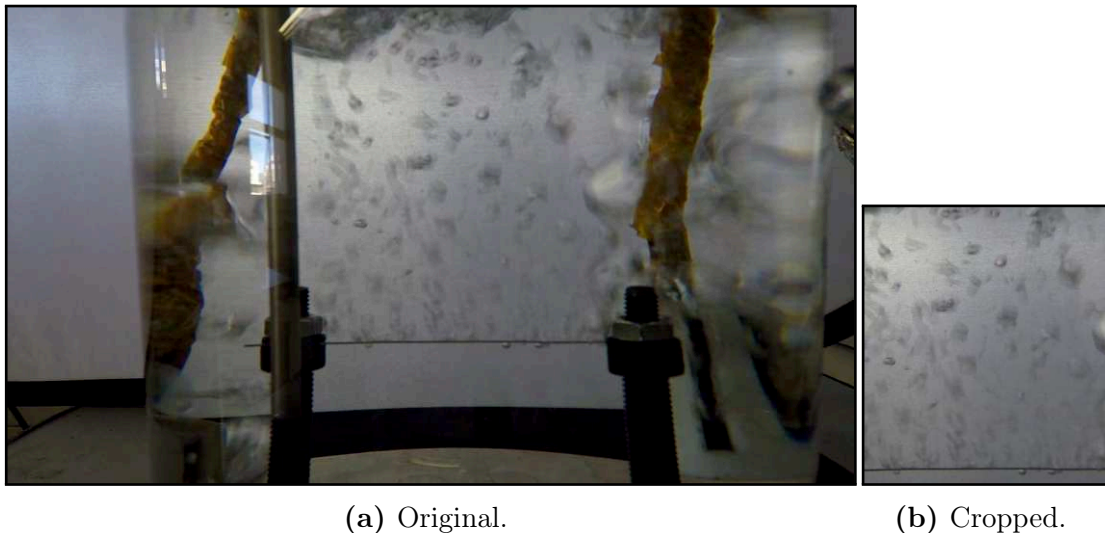


Figure 18 – Demonstration of the preprocessing step of cropping an image to the ROI. Obtained from the large wire pool boiling dataset at the power level of 30 W.

The motivation for this preprocessing step is manifold since it: (a) reduces the storage footprint of caching steps; (b) prevents the model from using environmental clues unrelated to the phase-change phenomenon, such as the background and the auxiliary heaters, to make predictions; (c) speeds up the preprocessing and training pipelines;

Table 7 – Image shapes before and after the ROI cropping preprocessing step for each pool boiling dataset. Shapes are presented in the format height \times width \times color channels.

Dataset	Image shape before	Image shape after
Large wire	$1520 \times 2704 \times 3$	$970 \times 855 \times 3$
Small wire	$1520 \times 2704 \times 3$	$870 \times 790 \times 3$
Horizontal ribbon	$1520 \times 2704 \times 3$	$850 \times 790 \times 3$
Vertical ribbon	$1520 \times 2704 \times 3$	$900 \times 830 \times 3$

(d) reduces the model size; and (e) introduces domain knowledge by explicitly telling the algorithms that the only region that matters is the ROI and nothing outside it.

4.2.2.3 Grayscaleing

Grayscaleing consists in converting a colored image to a scale of gray. This process combines the original image's red, green, and blue channels using a weighted average, as illustrated in Figure 19, resulting in a three-fold dimensionality reduction. The underlying assumption behind grayscaleing is that color is not an essential feature of phase-change phenomena and hence can be safely discarded without compromising model performance. Figure 20 supports this claim since there is little perceived difference between the colored and grayscale images. Table 8 presents the image shapes before and after grayscaleing.

$$\left(\begin{array}{c} \text{Red channel} \\ \left[\begin{array}{ccc} 215 & 72 & 8 \\ 202 & 69 & 152 \\ 113 & 233 & 126 \end{array} \right] \\ \text{Green channel} \\ \left[\begin{array}{ccc} 192 & 169 & 248 \\ 174 & 91 & 11 \\ 230 & 226 & 87 \end{array} \right] \\ \text{Blue channel} \\ \left[\begin{array}{ccc} 125 & 130 & 105 \\ 98 & 75 & 231 \\ 171 & 242 & 103 \end{array} \right] \end{array} \right)$$

↓ grayscale

$$\left[\begin{array}{ccc} 191.22 & 135.54 & 159.93 \\ 173.69 & 82.59 & 78.22 \\ 188.28 & 229.89 & 100.47 \end{array} \right]$$

Figure 19 – Demonstration of the preprocessing step of grayscaleing an image in matrix representation. The input image is encoded in 8-bit unsigned integer type and its color channels are shown as separate matrices for easier understanding.

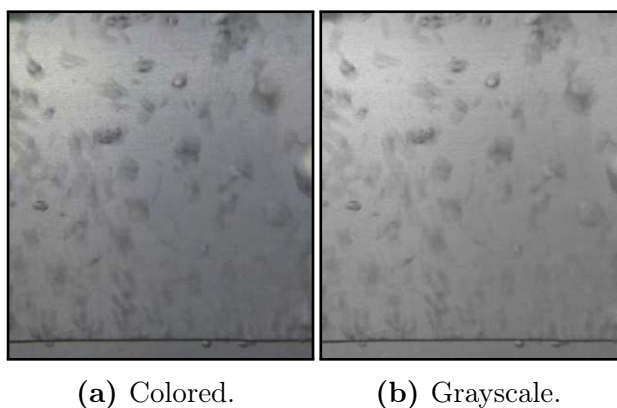


Figure 20 – Demonstration of the preprocessing step of grayscaleing an image. Obtained from the large wire pool boiling dataset at the power level of 30 W. There is little perceivable difference between (a) the colored and (b) the grayscale images.

Table 8 – Image shapes before and after the grayscaling preprocessing step for each pool boiling dataset. Shapes are presented in the format height \times width \times color channels.

Dataset	Image shape before	Image shape after
Large wire	$970 \times 855 \times 3$	$970 \times 855 \times 1$
Small wire	$870 \times 790 \times 3$	$870 \times 790 \times 1$
Horizontal ribbon	$850 \times 790 \times 3$	$850 \times 790 \times 1$
Vertical ribbon	$900 \times 830 \times 3$	$900 \times 830 \times 1$

4.2.2.4 Downscaling

Downscaling reduces the number of pixels in an image by locally averaging neighbor pixels. Each element of the output image is the mean of the elements in the corresponding window in the input image. For instance, Figure 21 illustrates a 2×2 downscaling operation.

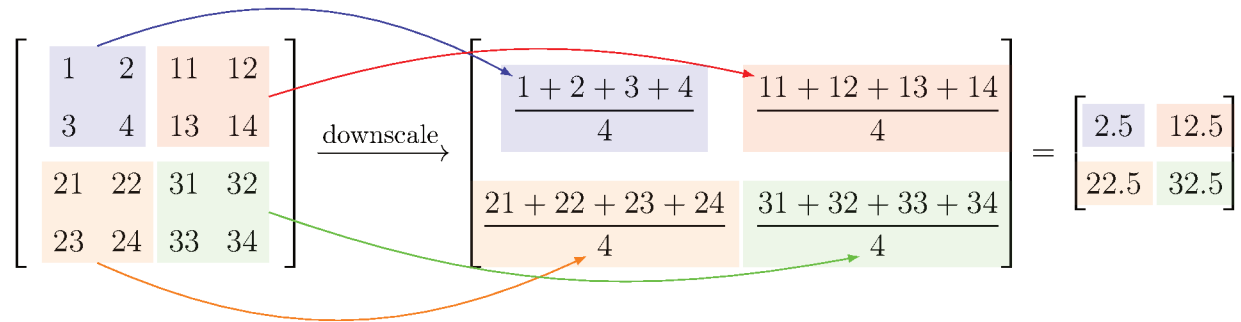


Figure 21 – Demonstration of the preprocessing step of downscaling an image with a downscaling factor of $f_{ds} = 2$, resulting in a $f_{ds}^2 = 4$ dimensionality reduction. Each pixel in the output image is the average of the elements in the corresponding 2×2 window in the input image.

Downscaling images in $f_{ds} \times f_{ds}$ windows, where f_{ds} is the *downscaling factor*, results in a dimensionality reduction of f_{ds}^2 . The benefits of image downscaling include (a) a significant speedup in the preprocessing and training pipelines; (b) smaller storage and RAM footprints of the dataset; and (c) noise removal, since downscaling acts as a filter for high-frequency noise in images. On the other hand, downscaling can also aggressively reduce the amount of information in the image, removing the learning substrate for training models. Hobold and da Silva (2018b, 2019b) analyzed the effects of downscaling images on the relative variance and cross-entropy ratio between the original and the downscaled images for different downscaling factors. They concluded that a downscaling factor of up to $f_{ds} = 6$ preserves most of the information in the original image, and chose $f_{ds} = 5$ in their preprocessing pipeline.

Following the procedure proposed by Hobold and da Silva (2018b, 2019b), the relative variance and cross-entropy ratio are computed from the 100-bin luminance distribution histograms HIST of frames. With this definition, the cross-entropy S between a

frame \mathbf{x} and its downsampled version $D^{f_{\text{ds}}}(\mathbf{x})$ is calculated as

$$S(\mathbf{x}, D^{f_{\text{ds}}}(\mathbf{x})) = \text{HIST}(\mathbf{x}) \cdot \ln \text{HIST}(D^{f_{\text{ds}}}(\mathbf{x})) = \sum_{j=0}^{100} \text{HIST}_j(\mathbf{x}) \ln \text{HIST}_j(D^{f_{\text{ds}}}(\mathbf{x})), \quad (18)$$

and their relative cross-entropy S^* is defined as

$$S^*(\mathbf{x}, D^{f_{\text{ds}}}(\mathbf{x})) := \frac{S(\mathbf{x}, D^{f_{\text{ds}}}(\mathbf{x}))}{S(\mathbf{x}, \mathbf{x})} = \frac{\text{HIST}(\mathbf{x}) \cdot \ln \text{HIST}(D^{f_{\text{ds}}}(\mathbf{x}))}{\text{HIST}(\mathbf{x}) \cdot \ln \text{HIST}(\mathbf{x})}. \quad (19)$$

Analogously, the relative variance VAR^* between a frame and its downsampled version can be calculated from the ratio of their variances VAR as in

$$\text{VAR}^*(\mathbf{x}, D^{f_{\text{ds}}}(\mathbf{x})) := \frac{\text{VAR}(D^{f_{\text{ds}}}(\mathbf{x}))}{\text{VAR}(\mathbf{x})} = \frac{\sum_{j=0}^{100} [\text{HIST}_j(D^{f_{\text{ds}}}(\mathbf{x})) - \overline{\text{HIST}}(D^{f_{\text{ds}}}(\mathbf{x}))]^2}{\sum_{j=0}^{100} [\text{HIST}_j(\mathbf{x}) - \overline{\text{HIST}}(\mathbf{x})]^2}. \quad (20)$$

Figure 22 presents the relative variance and cross-entropy ratio as functions of the downscaling factor obtained in this work for the four pool boiling datasets. Images were preprocessed utilizing $f_{\text{ds}} = 1-10$, and the relative variance and cross-entropy ratio to the original, non-downsampled image were computed. Similarly to what was observed by Hobold and da Silva (2018b), for downscaling factors greater than 6, both metrics increase sharply, indicating a significant change in the statistical properties of the image. For downscaling factors smaller than 6, the relative variance and cross-entropy ratio are relatively stable, indicating that the information contained in the image is preserved.

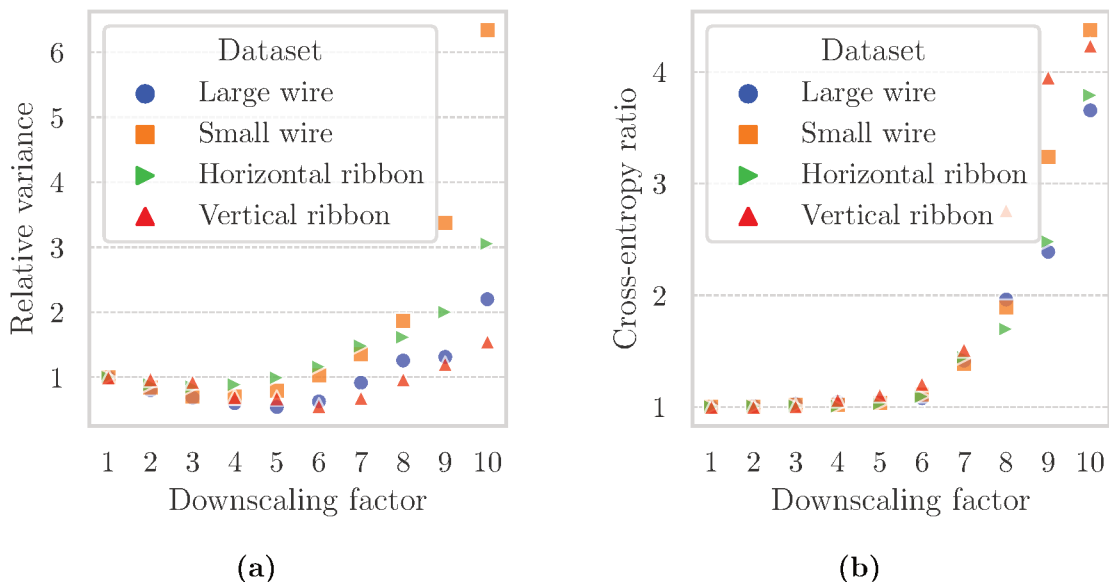


Figure 22 – Downscaling analyses of (a) the relative variance and (b) the cross-entropy ratio as functions of the downscaling factor f_{ds} for the four pool boiling datasets.

In addition to the retained information analysis, this work investigated the generalization error of trained models as a function of the downscaling factor in Section 5.1.2.

Based on those results and the analysis presented in Figure 22, this Thesis employs a downscaling factor of $f_{ds} = 5$, effectively reducing the dataset dimensionality by 25 times. Figure 23 demonstrates the application of the downscaling preprocessing step to a pool boiling frame. Table 9 reports the image shapes before and after this step.

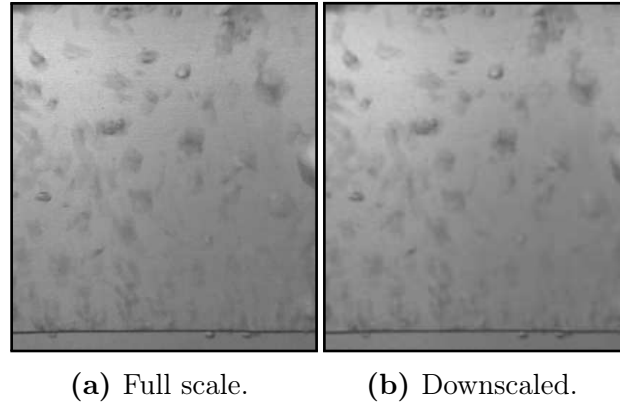


Figure 23 – Demonstration of the preprocessing step of downscaling an image using a downscaling factor of $f_{ds} = 5$. Obtained from the large wire pool boiling dataset at the power level of 30 W. There is little perceivable difference between (a) the full-scale and (b) the downscaled images, but the dataset dimensionality is reduced in $f_{ds}^2 = 25$ times.

Table 9 – Image shapes before and after downscaling each pool boiling dataset by a factor of $f_{ds} = 5$. Shapes are presented in the format height \times width \times color channels.

Dataset	Image shape before	Image shape after
Large wire	$970 \times 855 \times 1$	$194 \times 171 \times 1$
Small wire	$870 \times 790 \times 1$	$174 \times 158 \times 1$
Horizontal ribbon	$850 \times 790 \times 1$	$170 \times 158 \times 1$
Vertical ribbon	$900 \times 830 \times 1$	$180 \times 166 \times 1$

4.2.2.5 Size uniformization

As described in Sections 3.3 and 3.4, each experimental case includes multiple independently recorded videos. Because of variations in the positioning of the camera and other objects in the boiling chamber, such as the auxiliary heaters, the ROI for each video may be different from the others, even if only slightly. Consequently, the ROI cropping step described in Section 4.2.2.2 results in frames with different shapes, which propagates down the preprocessing pipeline.

However, standard deep-learning models require their inputs to have the same shape, including image datasets. Therefore, the size uniformization step ensures that all images belonging to the same dataset have the same height and width via center cropping, a preprocessing method in which images are symmetrically cropped at their borders,

keeping their centers invariant. Hobold and da Silva (2018b,a, 2019a,b) also utilized this process.

According to Goodfellow, Bengio, and Courville (2016), some machine learning models do not require image resizing; for instance, some models accept variably-sized inputs while keeping the output shape constant, whereas some other convolutional models have variable-sized outputs that scale with the inputs. Exploring such models, however, is out of the scope of the current study, and hence enforcing uniform shapes is regarded as a necessary step.

Size uniformization also provides an opportunity to meet hardware requirements for maximum efficiency. For instance, NVIDIA Tensor Cores enable performance boosts when training machine learning models on supported GPUs, as is the case in this work, as long as image dimensions are multiples of 8 (NVIDIA, 2022). Consequently, all images are center-cropped to the largest multiples of 8 that fit internally in the entire dataset.

In this work, the size uniformization preprocessing step is applied together with the visualization cropping. For this reason, the processed image shapes and an example are only shown in Section 4.2.2.6.

4.2.2.6 Vertical visualization cropping

In the pool boiling datasets, it is possible to differentiate two subcases: *direct* and *indirect* visualization, illustrated in Figure 24. This terminology follows Hobold and da Silva (2018b,a, 2019a,b). In direct visualization, the ROI contains the heater surface. In principle, direct visualization allows models to base their predictions on the behavior of bubbles before and during departure and on the visible characteristics of nucleation sites. On the other hand, in indirect visualization, an additional cropping step removes the heater surface from the images. In this situation, predictions are based only on the characteristics of bubbles after their departure, including how they interact in the liquid as they rise to the free surface. According to Hobold and da Silva (2018b), two main reasons justify the study of indirect visualization in addition to direct visualization: first, in some applications, the heater surface is not visible, and hence indirect observation is the only option; second, direct visualization might induce non-negligible bias in the predictions.

The direct and indirect visualization modes correspond to two datasets generated from the same source. In this work, the direct visualization dataset is the source dataset, and only height uniformization is performed in the visualization cropping step. The indirect visualization dataset, in its turn, is generated from the direct visualization dataset by cropping images at the bottom.

In order to keep consistency with Hobold and da Silva (2019b) and simultaneously keep shapes as multiples of 8, as justified in Section 4.2.2.5, all pool boiling frames are vertically cropped to 120 p in height, in direct visualization, and 72 p, in indirect visualization. Table 10 presents each dataset’s final image shapes in direct and indirect

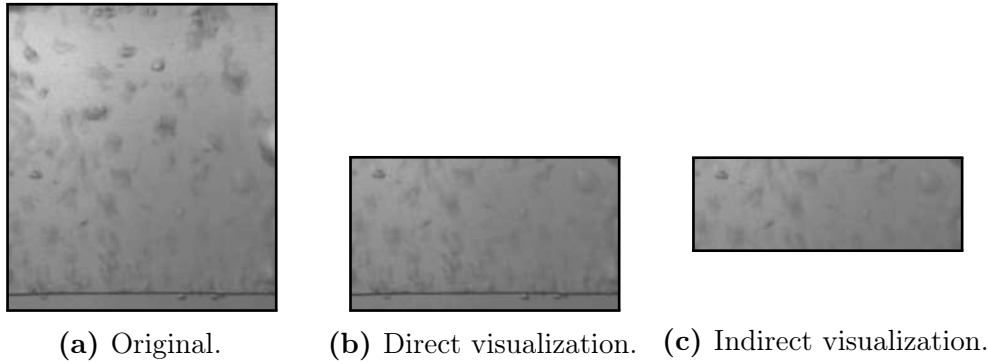


Figure 24 – Demonstration of the preprocessing step of size uniformization and visualization cropping, either (b) keeping the heater surface in the pool boiling images in direct visualization; or (c) removing it in indirect visualization.

visualization.

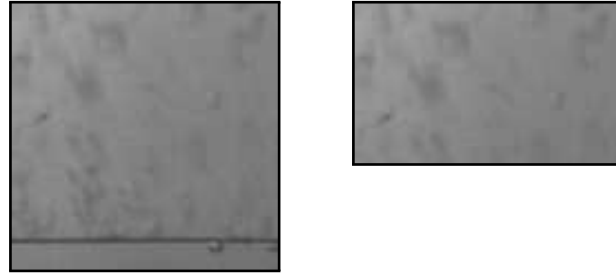
Table 10 – Image shapes before and after height uniformization and vertical crop in direct and indirect visualization. Shapes are presented in the format height \times width \times color channels.

Dataset	Image shape before	Direct visualization	Indirect visualization
Large wire	$194 \times 171 \times 1$	$120 \times 171 \times 1$	$72 \times 171 \times 1$
Small wire	$174 \times 158 \times 1$	$120 \times 158 \times 1$	$72 \times 158 \times 1$
Horizontal ribbon	$170 \times 158 \times 1$	$120 \times 158 \times 1$	$72 \times 158 \times 1$
Vertical ribbon	$180 \times 166 \times 1$	$120 \times 166 \times 1$	$72 \times 166 \times 1$

4.2.2.7 Horizontal visualization window cropping

Aiming to reduce data dimensionality further, Hobold and da Silva (2019b) performed a horizontal visualization window analysis by decreasing frame width via center cropping and evaluating trained models on them. Even though reducing the horizontal visualization window reduced model performance for all visualization fractions, the performance hit was insignificant for visualization windows larger than 50% of the original frame. Based on that, they employed visualization windows with 60% of the input width.

Section 5.1.3 evaluates the visualization window size's effects on the models' performance. Based on those results, the analyses from Hobold and da Silva (2019b), and to keep consistency with their results, this work also utilizes 60% of the visualization window of 196 p. As a result, this final cropping step uniformizes all frames to 120 p in width after rounding $60\% \times 196$ p to the nearest multiple of 8. Consequently, the final shape of the pool boiling images is uniformly equal to $120 \times 120 \times 1$, in direct visualization, and to $72 \times 120 \times 1$, in indirect visualization.



(a) Direct visualization. (b) Indirect visualization.

Figure 25 – Demonstration of the preprocessing step of horizontal visualization window cropping for (a) direct and (b) indirect visualization.

4.2.2.8 Standardization

According to Goodfellow, Bengio, and Courville (2016), either image normalization or standardization are frequently necessary since datasets containing images of varied pixel intensity ranges usually lead to training failure. In standardized images, the distribution of pixel luminance is equalized across all images in the dataset, which is beneficial to the optimization algorithm.

Images are standardized by moving their mean to 0 and their contrast to 1 via a linear transformation. Because of this, standardized images are more consistent with one another and tend to provide an easier learning landscape for model training. The default configurations of optimizers generally assume standard datasets (GOODFELLOW; BENGIO; COURVILLE, 2016); hence, standardized images simplify the search for optimal training hyperparameters. Moreover, image standardization also exempts the need for image brightness and contrast augmentation since any changes to those values are reverted during the standardization step.

A grayscale image represented by a matrix \mathbf{I} is standardized by subtracting its mean and dividing by its standard deviation, as in

$$\mathbf{I}_{\text{standardized}} = \frac{\mathbf{I} - \bar{\mathbf{I}}}{\sqrt{\text{VAR}(\mathbf{I})}}, \quad (21)$$

and as illustrated in Figure 26.

The standardization study presented in Section 5.1.1 demonstrates that image standardization significantly improves model performance with a negligible additional computational cost. For this reason, unless specifically stated the contrary, all dataset images in this work are normalized before being fed to machine learning models.

For comparison, Hobold and da Silva (2019b) and Scariot (2019) did not apply image standardization but only normalization, a process that scales pixel luminosity to a different range, generally $[0, 1]$. In this Thesis, normalization is accomplished during the image pixel format conversion preprocessing step, described in Section 4.2.2.1.

$$\begin{array}{ccc}
 \begin{bmatrix} 81 & 127 & 127 \\ 0 & 187 & 50 \\ 34 & 12 & 207 \end{bmatrix} & \xrightarrow{\text{standardize}} & \begin{bmatrix} -0.32 & 0.35 & 0.35 \\ -1.50 & 1.23 & -0.77 \\ 0.46 & -1.33 & 1.52 \end{bmatrix} \\
 \bar{\mathbf{I}} = 102.78 & & \bar{\mathbf{I}}_{\text{standardized}} = 0 \\
 \text{VAR}(\mathbf{I}) = 68.45 & & \text{VAR}(\mathbf{I}_{\text{standardized}}) = 1
 \end{array}$$

Figure 26 – Demonstration of the preprocessing step of standardizing images. In this example, a 3×3 grayscale image is standardized so that its mean and variance equal 0 and 1, respectively.

4.2.3 Dataset subsampling

Subsampling a dataset involves selecting only a subset of the data points from the original dataset, reducing the size of the dataset fed to machine learning models. In addition, subsampling can be applied to highly-correlated datasets to reduce data correlation (HOBOLD; DA SILVA, 2018b). Highly correlated data breaks the i.i.d. assumptions^[1] about the data samples, thus reducing the statistical significance of the results.

For video data, as is the case in this work, data correlation manifests as a high structural similarity (WANG et al., 2004) between consecutive video frames. Figure 27 shows the calculated structural similarity index between a reference frame, identified as frame 0, and its subsequent frames in all four pool boiling datasets. As shown, there is no distinct pattern in the structural similarity index distribution, which indicates that the frames are uncorrelated. Those findings agree with Hobold and da Silva (2018b).

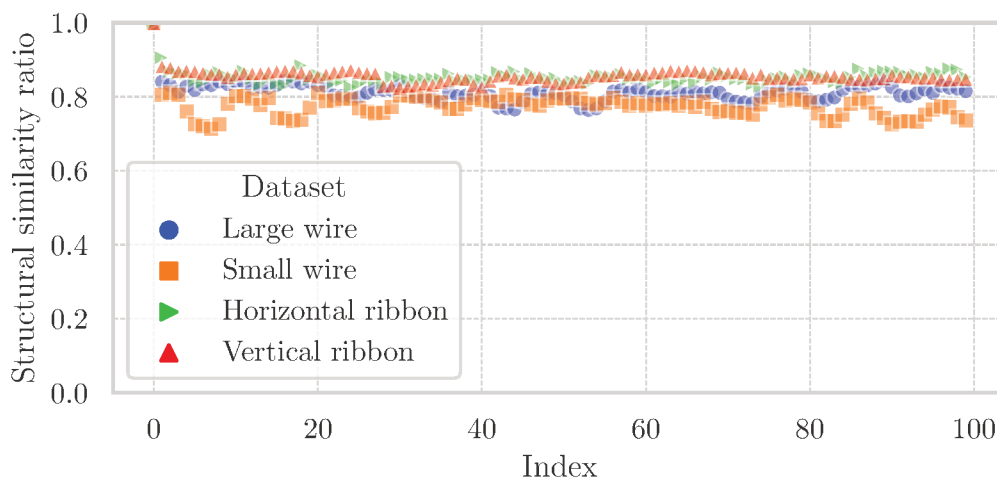


Figure 27 – Structural similarity between consecutive frames for the four pool boiling datasets. Generated by calculating the structural similarity index between a reference frame, identified as frame 0, and the subsequent 100 frames.

^[1] The i.i.d. assumptions require that all dataset examples are independent of each other and drawn from the same probability distribution (GOODFELLOW; BENGIO; COURVILLE, 2016).

The results from Figure 27 indicate that subsampling is unnecessary for the datasets employed in this work. This is also confirmed by the learning curve results from Section 5.1.4, which show that models benefit the most from training on the full dataset.

4.3 TRAINING

With the datasets well defined and preprocessed, a training pipeline is established to train the machine learning models according to the concepts explained in Section 2.2. The training pipeline consists of:

- **Datasets:** each training session requires a dataset to train models. Most datasets in this work are generated by submitting one of the experimental datasets obtained in Chapter 3 to the preprocessing pipeline. For each study, a different preprocessing configuration may be chosen to evaluate the effect of the preprocessing steps on the model performance, as done in Section 5.1. Additionally, the datasets are split into training, validation, and test sets, as described in Section 4.2.1.
- **Model architecture:** models are instantiated from an architecture, which defines the type, number, configuration, and connectivity of layers. All models trained in this work are CNNs containing a sequence of convolutional, activation, and pooling layers, topped by hidden dense layers and an output layer. Since this work focuses on the regression problem of quantifying a single scalar value, the output layer contains a single unit with a linear activation function. Except for the architectures generated by the AutoML pipeline explained in Section 4.5, this work employs the same architecture as Hobold and da Silva (2019b), illustrated in Figure 10.
- **Optimization algorithm:** models are trained by the Adam (KINGMA; BA, 2014) optimizer with a learning rate of 1×10^{-3} , the same as Hobold and da Silva (2019b) and Scariot (2019). Data is fed to the models in batches of 200 samples for consistency with Hobold and da Silva (2019b).
- **Performance metrics:** the MSE is chosen as the loss function optimized by the Adam optimizer. This is in alignment with Hobold and da Silva (2019b) and Scariot (2019) and with the recommendation by Goodfellow, Bengio, and Courville (2016) for deep learning regression problems. The other performance metrics described in Section 2.2.3 track the model performance on the validation set during training.
- **Early stopping:** models are trained for up to 100 epochs, but training stops earlier if the loss over the validation set stops improving for 10 consecutive epochs. After training ends, the model with the lowest loss over the validation

set is selected as the best model. This helps prevent overfitting since the best model is selected based on the validation loss, not the training loss.

4.4 VALIDATION

This Section presents machine learning results obtained to validate the training pipeline against Hobold and da Silva (2019b). This reference is chosen for validation because it utilized the same experimental setup and CNN architecture as this work. The validation presented in this Section consists of training models on the baseline, large wire dataset \mathcal{D}^{LW} , calculating the performance metrics, and comparing them with the values reported by Hobold and da Silva (2019b) and transcribed into Table 1. The entire pipeline described in this Chapter is considered validated if the metrics are similar.

Table 11 presents the results of the validation study. The results are presented as functions of the visualization mode (direct or indirect, as detailed in Section 4.2.2.6) and the evaluation subset (training, validation, or test). The reference results from Hobold and da Silva (2019b) are also displayed for comparison. In order to ensure a proper comparison, the results are presented only for the nucleate and film boiling regimes, defined by $q'' \geq 10 \text{ W/cm}^2$.

The results in Table 11 demonstrate that the training pipeline employed in this work achieves equivalent performance to the reference, especially if the uncertainty intervals are considered. Additionally, the differences between the results can be attributed to natural differences in the experimental setup. In particular, as shown in Section 5.2, model performance is significantly affected by the heater surface, and metrics obtained by training and testing a model on one surface do not necessarily generalize to other surfaces. Since the large wire dataset \mathcal{D}^{LW} utilized in this work was obtained from a different heater surface than the one utilized by Hobold and da Silva (2019b), differences in the results are expected.

The validation results also provide insight into other aspects of the training pipeline. For example, the results demonstrate that the trained model correctly generalizes to unseen data. In direct visualization, the validation MSE is only approximately 27% higher than the training MSE. The difference is even less significant when comparing metrics such as the RMSE or MAPE. Indirect visualization results in even better generalization capabilities, with the validation MSE being only approximately 14% higher than the training MSE. This is expected since the indirect visualization is less informative than the direct visualization; hence, the model is forced to learn more generalizable features. According to Goodfellow, Bengio, and Courville (2016), a gap between the training and validation metrics is always expected, even for models that generalize well, as in this case.

It can also be observed that the direct visualization mode achieves better results than the indirect visualization mode, a similar finding to Hobold and da Silva (2019b). This is expected since the direct visualization contains more information than the indirect

Table 11 – Training pipeline validation results. Metrics are presented only for nucleate and film boiling regimes, defined by $q'' \geq 10 \text{ W/cm}^2$. Metrics for the training, validation and test sets are presented as functions of the type of visualization: direct, in which the heater surface is visible in the images, or indirect, in which it is omitted. Confidence intervals of 95 % around the mean were computed by bootstrapping 1000 batches of 1000 samples. The reference results are reproduced from Hobold and da Silva (2019b).

Metric	Unit	This work			Reference	
		Training	Validation	Test	Validation	Test
Direct						
MSE	$(\text{W/cm}^2)^2$	$12.32_{-0.06}^{+0.06}$	$15.69_{-0.06}^{+0.06}$	$15.41_{-0.03}^{+0.03}$	13_{-1}^{+1}	13_{-1}^{+1}
RMSE	W/cm^2	$3.507_{-0.009}^{+0.008}$	$3.959_{-0.008}^{+0.008}$	$3.925_{-0.004}^{+0.004}$	—	—
MAE	W/cm^2	$2.720_{-0.007}^{+0.006}$	$3.053_{-0.006}^{+0.006}$	$3.025_{-0.003}^{+0.003}$	$2.8_{-0.1}^{+0.1}$	3_{-3}^{+6}
MAPE	%	$6.97_{-0.02}^{+0.02}$	$7.62_{-0.02}^{+0.02}$	$7.570_{-0.008}^{+0.008}$	$7.4_{-0.4}^{+0.5}$	$7.6_{-0.4}^{+0.4}$
R^2	—	$0.9688_{-0.0002}^{+0.0002}$	$0.9601_{-0.0002}^{+0.0002}$	$0.96090_{-0.00008}^{+0.00008}$	$0.983_{-0.002}^{+0.002}$	$0.983_{-0.002}^{+0.002}$
Indirect						
MSE	$(\text{W/cm}^2)^2$	$29.60_{-0.10}^{+0.10}$	$33.84_{-0.05}^{+0.05}$	$33.3_{-0.1}^{+0.1}$	34_{-4}^{+4}	33_{-4}^{+4}
RMSE	W/cm^2	$5.439_{-0.009}^{+0.009}$	$5.816_{-0.004}^{+0.004}$	$5.76_{-0.01}^{+0.01}$	—	—
MAE	W/cm^2	$4.274_{-0.006}^{+0.006}$	$4.475_{-0.005}^{+0.005}$	$4.530_{-0.009}^{+0.009}$	4_{-4}^{+10}	4_{-4}^{+10}
MAPE	%	$10.80_{-0.02}^{+0.01}$	$11.07_{-0.01}^{+0.01}$	$11.38_{-0.02}^{+0.02}$	$10.6_{-0.6}^{+0.6}$	$10.3_{-0.5}^{+0.5}$
R^2	—	$0.9250_{-0.0003}^{+0.0003}$	$0.9141_{-0.0002}^{+0.0002}$	$0.9156_{-0.0004}^{+0.0004}$	$0.956_{-0.005}^{+0.005}$	$0.956_{-0.005}^{+0.005}$

visualization, especially regarding the boiling behavior near the heater surface. Because of this, models trained on direct visualization learn more specific features and hence achieve lower error metrics.

In addition, the validation and test metrics are very close, indicating that error metrics evaluated on the validation set are a precise estimate for the generalization error represented by the test metrics.

The errors in Table 11 also highlight CNNs as a promising tool for estimating heat flux in pool boiling compared to the numerical correlations presented in Section 2.1.3. The best correlation found for quantifying the pool boiling of water is given by Equation (5) with an associated error of 4.82 %. In addition, the correlation requires the measurement of the temperature at the heater surface, which is not always possible, and might incur a significant increase in the global uncertainty of the predicted values after composing the correlation uncertainty with the measurement uncertainty. For comparison, the CNNs trained in this Section achieved a global error of 7.62 % on the validation set in direct visualization and 11.07 % in indirect visualization.

The datasets constructed in this Section and the CNNs trained are referred to in

the following Sections as *baseline datasets* and *baseline models*, respectively, since they represent the best results obtained in previous studies. The new results presented in Chapter 5 are compared to them to evaluate their performance.

4.5 AUTOMATED MACHINE LEARNING

Section 5.3 presents a novel study about the application of AutoML for searching optimal architectures of CNNs to estimate heat transfer in pool boiling. The present Section describes the AutoML pipeline and the decisions made during its design.

AutoML is managed in this work by AutoKeras (JIN, Haifeng; SONG; HU, 2019) utilizing the greedy search algorithm, which, as explained in Section 2.2.4, separates hyperparameters into different categories. At each trial, the algorithm selects one of the categories and randomly samples hyperparameters from it. The hyperparameters from the other categories are fixed and equal to the best options found until that trial. The search is terminated when the maximum number of 100 trials is reached.

At each trial of the AutoML pipeline, the training pipeline described in Section 4.3 is executed once with a few modifications, including the choice of the optimizer algorithm and the model architecture. The entire AutoML pipeline can be summarized as:

- **Datasets:** in the AutoML studies, the datasets are preprocessed by the default pipeline detailed in Section 4.2.2 without any modifications. The best model is selected as the one that performs best on the validation set. Based on the results from Section 5.1.1, images are always standardized; hence, this choice is removed from the search space.
- **Model architecture:** the model architecture is part of the search space of the AutoML pipeline. Since the problem to be solved is a visualization-based regression, the search space is constrained to only convolutional networks with a single, linearly activated output unit. This way, the search space includes the number and configuration of convolutional layers, including their filter count, kernel sizes, activation function, and associated pooling operation. In addition, the search space also includes the choice between regular and depth-wise separable convolutions (CHOLLET, 2016) and the application of dropout (SRIVASTAVA et al., 2014). On top of the convolutional layers, the search space includes the number and configuration of dense layers, including their unit count, activation function, dropout rate, and the application of batch normalization (GOODFELLOW; BENGIO; COURVILLE, 2016).
- **Optimization algorithm:** instead of utilizing the Adam optimizer, the optimizer algorithm and associated configuration parameters, such as the learning rate, are included in the search space. The batch size during training is set to 32 instead of 200. This was necessary to avoid memory exhaustion during the

search. The value of 32 is the default value chosen by AutoKeras.

- **Performance metrics:** similarly to the training pipeline described in Section 4.3, the AutoML pipeline uses the MSE as the performance metric to be optimized on the training set. In addition, the MSE evaluated on the validation set is also utilized to choose the best model.
- **Early stopping:** differently from the regular training pipeline, in the AutoML pipeline models are allowed to train for up to 1000 epochs, and training is stopped when the loss over the validation set stops improving for 10 consecutive epochs. After the training ends, the model weights that resulted in the lowest loss over the validation set are selected as the best model.
- **Maximum model size:** the greedy algorithm can also be configured to avoid training models that are considered too big. This is an important option since, as pointed out in Section 2.3, industrial applications frequently require models that can be deployed on embedded devices with limited computational resources. Additionally, limiting the model size helps reduce the time complexity of training and evaluating multiple models, an essential consideration for the initial exploratory analysis proposed in this Thesis. Moreover, size-constrained architectural search incentivizes the search algorithm to try more efficient models instead of arbitrarily large ones. In this work, the maximum number of parameters allowed in a model is set to 23 041 233 in direct visualization and 13 825 233 in indirect visualization, corresponding to the sizes of the baseline models trained in Section 4.4. This way, the algorithm guarantees that the best model found will be, at most, as large as the baseline models and likely smaller.

The exact search space explored by the AutoML pipeline is presented in Table 12. This search space is generated by the default AutoKeras (JIN, Haifeng; SONG; HU, 2019) template for image regression problems after adding the constraints mentioned in this Section. The search space includes constructs described elsewhere in the literature, such as AdamW (LOSHCHILOV; HUTTER, 2019) and depthwise separable convolutions (CHOLLET, 2016). In addition, some hyperparameters can be duplicated and have their search spaces multiplied depending on the values of other hyperparameters: the number of units in each dense layer, for instance, can be different in each layer.

In order to simplify the search space and keep models within the size constraint, widely adopted pre-trained models such as ResNet (HE et al., 2015) and Xception (CHOLLET, 2016) were not included in this work. Instead, adding them to the search space is left as a suggestion for future work in Section 6.2.

Table 12 – Automated machine learning search space. Each category of hyperparameters is displayed separately. The search space of each hyperparameter is shown as a discrete set of the possible values it might assume, or the set {NO, YES} for configurations that might be absent or present, respectively.

Hyperparameter	Search space
<i>Architecture – Convolutional layers</i>	
Number of convolutional blocks	{ 1, 2, 3 }
Number of consecutive convolutional layers per block	{ 1, 2 }
Convolutional kernel size	{ 3, 5, 7 }
Number of filters per layer	{ 16, 32, 64, 128, 256, 512 }
Depthwise separable convolutions?	{ NO, YES }
Apply max-pooling?	{ NO, YES }
Dropout rate	{ 0, 25 %, 50 % }
<i>Architecture – Reduction layer</i>	
Spatial reduction layer type	{Flattening, Global average pooling, Global max-pooling}
<i>Architecture – Dense layers</i>	
Use batch normalization?	{ NO, YES }
Number of dense layers	{ 1, 2, 3 }
Number of units per layer	{ 16, 32, 64, 128, 256, 512, 1024 }
Dropout rate	{ 0, 25 %, 50 % }
<i>Optimizer</i>	
Optimizer algorithm	{ SGD, Adam, AdamW }
Learning rate	{ 1×10^{-1} , 1×10^{-2} , 1×10^{-3} , 1×10^{-4} , 2×10^{-5} , 1×10^{-5} }

5 RESULTS AND DISCUSSION

Given the objectives described in Section 1.1, this Chapter presents the results of the research conducted in this Thesis. The results are presented in the form of a series of studies, each described in a separate section, and applying the methodology described in Chapter 4.

In Section 5.1, the effects of the preprocessing pipeline on the performance of machine learning models are studied. Those results justify the choices when designing the preprocessing pipeline and act as a basis for the subsequent Sections. Section 5.2 investigates the performance of CNNs trained and evaluated on datasets gathered from different heater surfaces. This provides a better understanding of how models generalize from one surface to another. Finally, Section 5.3 explores the application of AutoML to search for an optimal architecture for the CNN models and demonstrates its potential to improve the performance of machine learning models.

5.1 DATASET PREPROCESSING

In data-centric problems such as machine learning, the definition of the dataset, and hence the preprocessing pipeline associated with it, is of utmost importance. This Section presents the results of the preliminary analyses made to optimize the preprocessing pipeline and to understand the impact of the dataset on the performance of trained machine learning models.

5.1.1 Image standardization

One of the preliminary analyses made to maximize the performance of machine learning models was the study of image standardization. As explained in Section 4.2.2.8, image standardization consists of transforming images so that the mean luminance of their pixels is 0, and the standard deviation is 1.

In this study, machine learning models were trained with standardized and non-standardized images in direct and indirect visualization, totaling four models. The baseline large wire dataset \mathcal{D}^{LW} was preprocessed using the pipeline described in Section 4.2.2, except that the standardization preprocessing step was skipped in the non-standardized cases. In the non-standardized case, *image normalization* was applied due to the image pixel format conversion described in Section 4.2.2.1. After training, each model was evaluated on the validation and test sets. Table 13 presents the results in terms of the same performance metrics reported by Hobold and da Silva (2019b) and reproduced in Section 2.3.

As shown in Table 13, image standardization significantly reduces the final validation error of models, particularly in the indirect visualization mode. The validation error

Table 13 – Image standardization performance metrics. Validation and test performance metrics of the baseline architecture trained with two versions of the default training pipeline: with and without the image standardization step. The metrics are presented for the direct and indirect visualization cases of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$.

Metric	Unit	Direct		Indirect	
		Validation	Test	Validation	Test
Normalized					
MSE	$(\text{W/cm}^2)^2$	$15.69^{+0.06}_{-0.06}$	$15.41^{+0.03}_{-0.03}$	$33.84^{+0.05}_{-0.05}$	$33.3^{+0.1}_{-0.1}$
RMSE	W/cm^2	$3.959^{+0.008}_{-0.008}$	$3.925^{+0.004}_{-0.004}$	$5.816^{+0.004}_{-0.004}$	$5.76^{+0.01}_{-0.01}$
MAE	W/cm^2	$3.053^{+0.006}_{-0.006}$	$3.025^{+0.003}_{-0.003}$	$4.475^{+0.005}_{-0.005}$	$4.530^{+0.009}_{-0.009}$
MAPE	%	$7.62^{+0.02}_{-0.02}$	$7.570^{+0.008}_{-0.008}$	$11.07^{+0.01}_{-0.01}$	$11.38^{+0.02}_{-0.02}$
R^2	—	$0.9601^{+0.0002}_{-0.0002}$	$0.96090^{+0.00008}_{-0.00008}$	$0.9141^{+0.0002}_{-0.0002}$	$0.9156^{+0.0004}_{-0.0004}$
Standardized					
MSE	$(\text{W/cm}^2)^2$	$9.64^{+0.03}_{-0.03}$	$9.37^{+0.03}_{-0.03}$	$18.08^{+0.03}_{-0.03}$	$18.14^{+0.04}_{-0.04}$
RMSE	W/cm^2	$3.103^{+0.005}_{-0.005}$	$3.060^{+0.004}_{-0.004}$	$4.252^{+0.004}_{-0.004}$	$4.258^{+0.005}_{-0.005}$
MAE	W/cm^2	$2.361^{+0.004}_{-0.005}$	$2.336^{+0.003}_{-0.003}$	$3.247^{+0.003}_{-0.003}$	$3.233^{+0.003}_{-0.003}$
MAPE	%	$5.70^{+0.01}_{-0.01}$	$5.633^{+0.007}_{-0.006}$	$7.707^{+0.008}_{-0.008}$	$7.69^{+0.01}_{-0.01}$
R^2	—	$0.97552^{+0.00008}_{-0.00008}$	$0.97623^{+0.00008}_{-0.00008}$	$0.9541^{+0.0001}_{-0.0001}$	$0.9539^{+0.0001}_{-0.0002}$

decreased by nearly 39 % for direct visualization and 47 % for indirect visualization. The difference between the validation errors of the two visualization modes also substantially decreased when images were standardized.

This result aligns well with what is recommended by the literature (GOODFELLOW; BENGIO; COURVILLE, 2016) and demonstrates how a simple preprocessing step can significantly improve model performance. Due to the resulting increase in performance with almost no extra computation, image standardization was added to the preprocessing pipeline utilized in subsequent studies.

The improvement in the generalization error when standardizing images can be understood by inspecting the brightness and contrast in a dataset. Figure 28 displays a letter-value plot (HOFMANN; KAFADAR; WICKHAM, 2011; WASKOM, 2021) of the image brightness in direct visualization, calculated as the mean luminance of the pixels in the image, as a function of the nominal thermal power. Results in indirect visualization are almost equal. For each nominal power level, image brightness values are presented for the training, validation, and test sets. Outliers are discarded for simplification. Figure 28 demonstrates that the image brightness distributions vary depending on the nominal power level and, to a smaller degree, the subset (training, validation, or test). In particular, apart from outliers, brightness is sufficient to distinguish the 0 W power level

from the 20 W. However, image brightness is not a property of the boiling phenomenon but a characteristic of the experimental procedure. By removing the image brightness variation, image standardization ensures that the model is not biased towards a particular brightness level and instead learns the correct relationships between the image and the heat flux. The same reasoning can be applied to image contrast.

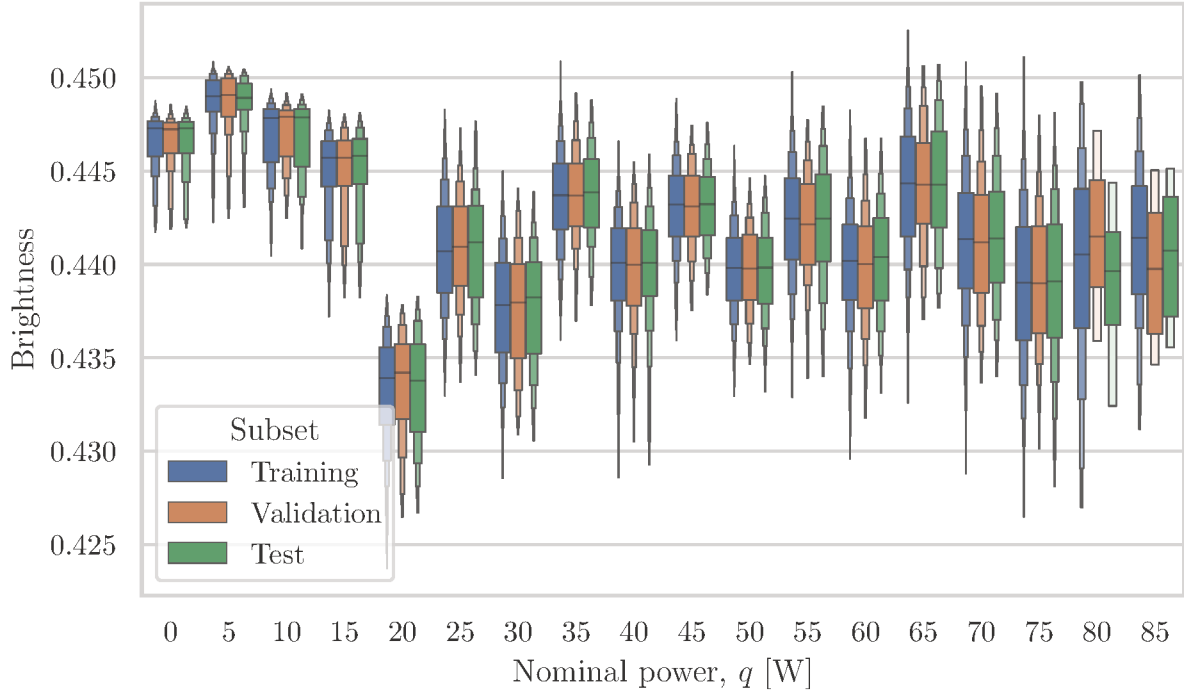


Figure 28 – Image brightness distribution in the baseline, large wire dataset in direct visualization. The result is presented as a letter-value plot (HOFMANN; KAFADAR; WICKHAM, 2011; WASKOM, 2021) with the removal of outliers. The distribution is shown for each nominal power level q for the training, validation and test subsets.

This study concludes that image standardization should be used to improve models’ generalization errors without significant extra computation. Hence all subsequent studies in this Thesis include the image standardization preprocessing step.

5.1.2 Image downscaling

Image downscaling is a preprocessing step that reduces the size of the input images by locally averaging neighbor pixels, as explained in Section 4.2.2.4. Downscaling inputs can improve the generalization error of trained models by reducing the amount of information they can access. In this section, the effect of image downscaling on the model’s generalization error is studied.

This study was conducted by training models with the baseline architecture illustrated in Figure 10. Models were trained on the baseline large wire dataset \mathcal{D}^{LW} preprocessed by the default preprocessing pipeline described in Section 4.2.2.4 with different

image downscaling factors in the downscaling preprocessing step. The downscaling factors studied were $f_{ds} = 1, 2, 3, 4, 5$ and 6. Larger downscaling factors were not considered since they resulted in images smaller than the required visualization windows described in Sections 4.2.2.5 to 4.2.2.7.

Figure 29 presents the results of this study. As shown, a general trend exists of decreasing validation error as the downscaling factor increases. This trend is particularly evident in the indirect visualization mode, where the validation error decreases by 69% when the downscaling factor is increased from $f_{ds} = 1$ to $f_{ds} = 5$ and by 74% when $f_{ds} = 6$. In the direct visualization mode, the validation error decreases by 66% when the downscaling factor is increased from $f_{ds} = 1$ to $f_{ds} = 5$ and by 59% when $f_{ds} = 6$.

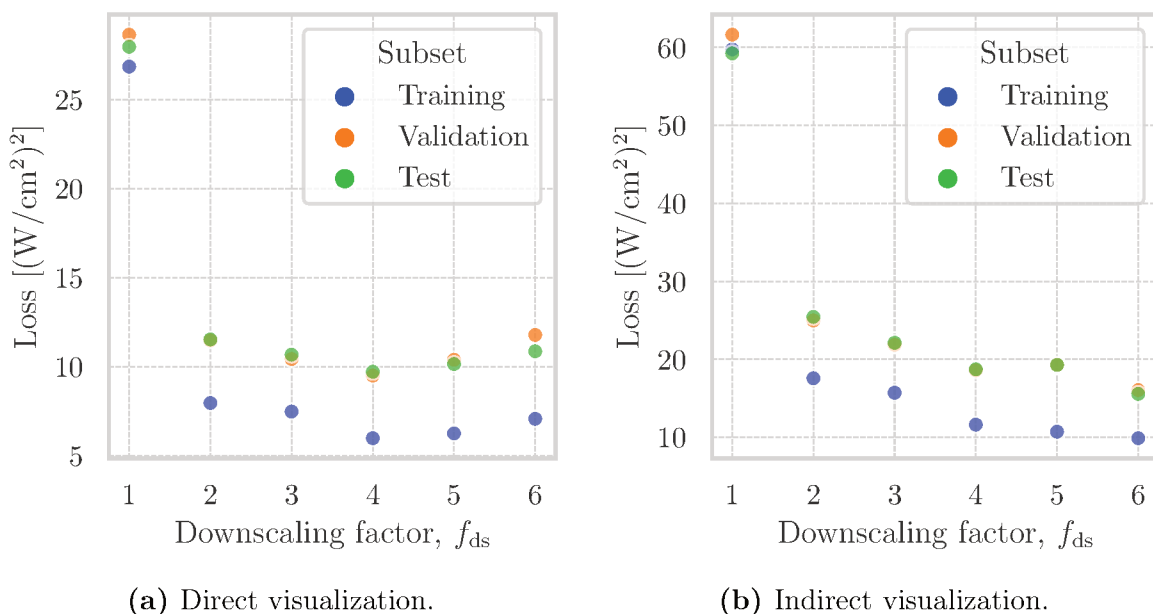


Figure 29 – Image downscaling results presented as the loss (MSE) seen as a function of the downscaling factor for the training, validation, and test subsets of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$.

Since downscaling is equivalent to passing images through an average pooling layer of size $f_{ds} \times f_{ds}$, the same reasons that explain why average pooling layers improve the generalization error of a model also apply to image downscaling. In particular, downscaling acts as a filter for high-frequency noise in images and helps models focus on the relevant features for predicting the heat flux.

It can also be observed that, for a downscaling factor of $f_{ds} = 6$, the generalization error increased in direct visualization relative to $f_{ds} = 5$ but not in indirect visualization. This is likely because the better performance in direct visualization mode depends on small-scale information present at the wire surface, which loses resolution when the downscaling factor is increased.

The choice of the downscaling factor for a given dataset is a trade-off between the computational cost of the model and the generalization error in the two visualization

modes. In order to keep consistency with the results of Hobold and da Silva (2019b), and since the increase in the generalization error in both visualization modes is not significant, the downscaling factor of $f_{\text{ds}} = 5$ was chosen in this work to integrate the default preprocessing pipeline.

5.1.3 Visualization window size

As shown by Hobold and da Silva (2019b), the visualization window size is a hyperparameter that affects the model’s generalization error. In their study, they demonstrated that, by reducing the size of the visualization window, the generalization error of the model increased. However, this increase was only significant for visualization fractions below 50%. For visualization fractions above 50%, the increase in the generalization error is not enough to justify the computational cost of training the model with a larger visualization window. Because of this, Hobold and da Silva (2019b) utilized a visualization window of 60% of the original frame width.

In this work, the visualization window size’s effect on the model’s generalization error is studied. The visualization window sizes studied were 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% and 100%. The visualization window size of 100% corresponds to the original frame width. A model was trained for each visualization window size, with the same architecture and hyperparameters as the baseline model described in Section 4.4. The models were trained on the baseline, large wire dataset \mathcal{D}^{LW} , preprocessed by the default preprocessing pipeline described in Section 4.2.2. Figure 30 presents the results from this study, showing the loss as a function of the visualization window fraction for the training, validation, and test subsets of \mathcal{D}^{LW} .

Figure 30 shows that a visualization window of the 60% the original frame width results in the lowest generalization error in both direct and indirect visualization, calculated over the validation set. This is different from what Hobold and da Silva (2019b) observed since, in their study, reducing the visualization window always deteriorated model performance. In Figure 30, there is a local optimum around 60%. Despite this, it is also possible to observe that the model’s generalization error is not significantly affected by the visualization window size.

Because of this result, and to keep consistency with the results of Hobold and da Silva (2019b), the visualization window size of 60% was chosen to integrate the default preprocessing pipeline.

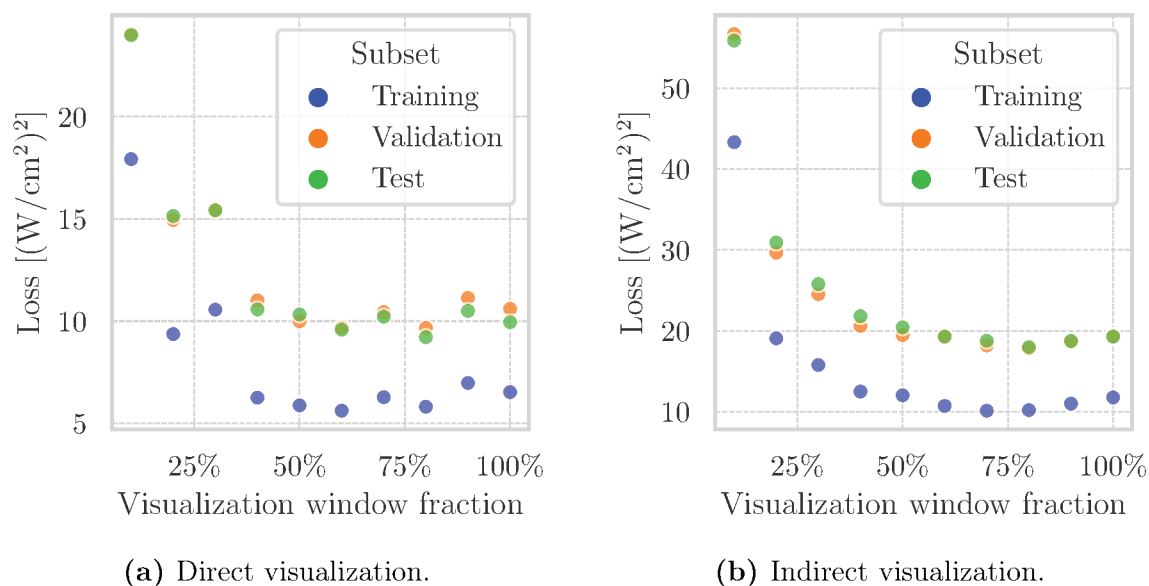


Figure 30 – Visualization window size results presented as the loss seen as a function of the visualization window fraction for the training, validation, and test subsets of the baseline, large wire dataset for $q'' \geq 10 \text{ W/cm}^2$. The results are shown for both (a) direct and (b) indirect visualization modes.

5.1.4 Learning curve

A *learning curve* of a model is a plot of its error as a function of the size of its training set^[1]. Intuitively, the more data a model consumes when training, the better it should score, as long as the dataset is sampled randomly from its source distribution (MURPHY, 2012). In addition, larger datasets help prevent overfitting and reduce the generalization error (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figure 31 presents the results of the learning curve study. The training set was subsampled with fractions of 1–10 % in steps of 1 % and 10–100 % in steps of 10 %. For each subsampled training set, a model was trained with the same architecture and hyperparameters as the baseline model. The models were trained on the subsampled large wire dataset \mathcal{D}^{LW} , preprocessed with the default preprocessing pipeline.

The results demonstrate that the generalization error reduces as the dataset subsample size increases, as expected. Despite this trend, the generalization error stabilizes for subsamples greater than or equal to 80 %, which suggests that those subsamples saturate the model capacity. Consequently, no benefits would be expected from gathering larger datasets. On the other hand, Figure 31 also shows that the generalization error significantly increases with reduced training set sizes, which motivates the choice of employing the entire training dataset in Section 4.2.3.

^[1] This definition follows Murphy (2012). Other authors define the learning curve as the model’s loss as a function of the training epoch (GOODFELLOW; BENGIO; COURVILLE, 2016).

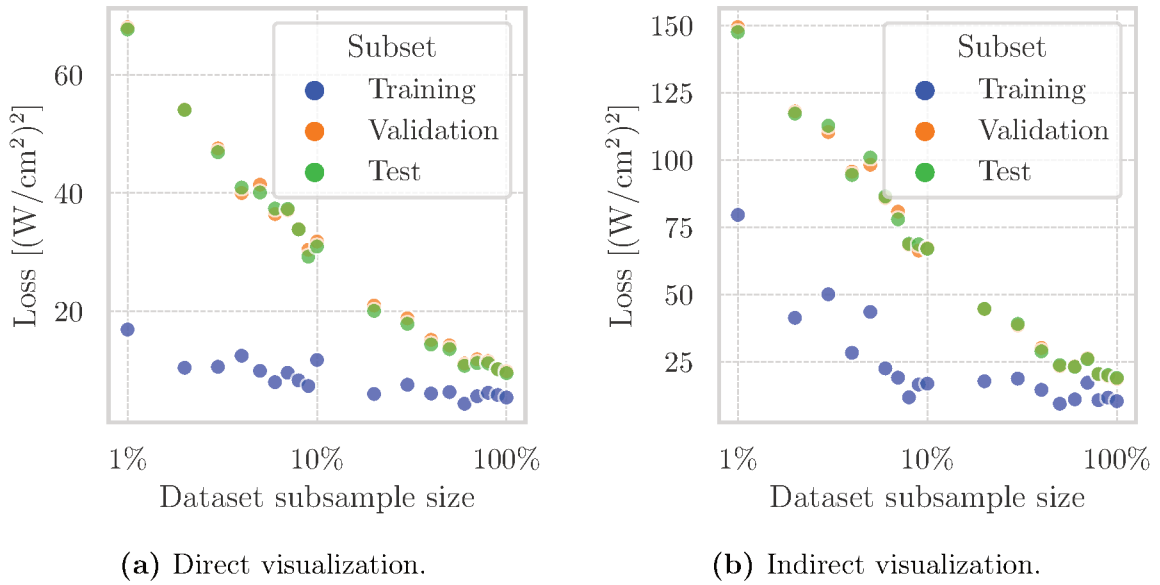


Figure 31 – Learning curve results presented as the loss seen as a function of the training dataset subsampling fraction for the training, validation, and test subsets of the baseline large wire dataset for $q'' \geq 10 \text{ W/cm}^2$. Training metrics are evaluated on the subsampled dataset, whereas validation and test metrics include 100% of the data. The results are shown for both (a) direct and (b) indirect visualization modes.

5.2 MULTIPLE SURFACES

The studies presented in Section 5.1 considered only the baseline, large wire dataset to optimize the preprocessing pipeline for model performance and generalization. However, one of this work’s objectives is to understand models’ ability to generalize in different contexts, especially on different heater surfaces. To achieve that goal, Section 5.2.1 studies models trained and evaluated on the same heater surface as a baseline result for Section 5.2.2, which studies models trained and evaluated on multiple heater surfaces.

5.2.1 Single-surface evaluation

In order to investigate the behavior of the validated preprocessing and training pipelines on different heater surfaces, this Section extends the work done in Section 4.4 by training and evaluating models on all four heater surfaces separately.

Tables 14 and 15 present the training, validation, and test metrics for models trained and evaluated on each heater surface separately in direct and indirect visualization, respectively. The metrics obtained in this study are comparable to or lower than the values reported by Hobold and da Silva (2019b) and reproduced in Table 11 in both direct and indirect visualization modes. This suggests that the preprocessing and training pipelines are robust to the different heater surfaces and achieve promising results under different operating conditions.

Table 14 – Training, validation, and test metrics for models trained and evaluated on each heater surface separately in direct visualization for $q'' \geq 10 \text{ W/cm}^2$.

Metric	Unit	Subset		
		Training	Validation	Test
Large wire				
MSE	$(\text{W/cm}^2)^2$	$5.43_{-0.04}^{+0.04}$	$9.64_{-0.03}^{+0.03}$	$9.37_{-0.03}^{+0.03}$
RMSE	W/cm^2	$2.327_{-0.009}^{+0.008}$	$3.103_{-0.005}^{+0.005}$	$3.060_{-0.004}^{+0.004}$
MAE	W/cm^2	$1.792_{-0.007}^{+0.006}$	$2.361_{-0.005}^{+0.004}$	$2.336_{-0.003}^{+0.003}$
MAPE	%	$4.48_{-0.02}^{+0.02}$	$5.70_{-0.01}^{+0.01}$	$5.633_{-0.006}^{+0.007}$
R^2	—	$0.9862_{-0.0001}^{+0.0001}$	$0.97552_{-0.00008}^{+0.00008}$	$0.97623_{-0.00008}^{+0.00008}$
Small wire				
MSE	$(\text{W/cm}^2)^2$	$11.89_{-0.04}^{+0.05}$	$18.15_{-0.05}^{+0.05}$	$19.37_{-0.03}^{+0.03}$
RMSE	W/cm^2	$3.446_{-0.006}^{+0.006}$	$4.259_{-0.006}^{+0.006}$	$4.401_{-0.003}^{+0.003}$
MAE	W/cm^2	$2.679_{-0.005}^{+0.005}$	$3.246_{-0.004}^{+0.004}$	$3.348_{-0.003}^{+0.002}$
MAPE	%	$7.23_{-0.01}^{+0.01}$	$8.209_{-0.010}^{+0.010}$	$8.238_{-0.005}^{+0.005}$
R^2	—	$0.9785_{-0.0001}^{+0.0001}$	$0.96724_{-0.00008}^{+0.00008}$	$0.96498_{-0.00005}^{+0.00005}$
Horizontal ribbon				
MSE	$(\text{W/cm}^2)^2$	$9.55_{-0.03}^{+0.03}$	$12.37_{-0.03}^{+0.03}$	$12.88_{-0.04}^{+0.04}$
RMSE	W/cm^2	$3.089_{-0.005}^{+0.005}$	$3.517_{-0.005}^{+0.005}$	$3.589_{-0.005}^{+0.005}$
MAE	W/cm^2	$2.359_{-0.003}^{+0.004}$	$2.660_{-0.003}^{+0.003}$	$2.715_{-0.004}^{+0.004}$
MAPE	%	$5.69_{-0.01}^{+0.01}$	$6.267_{-0.009}^{+0.009}$	$6.415_{-0.008}^{+0.007}$
R^2	—	$0.98056_{-0.00007}^{+0.00007}$	$0.97481_{-0.00008}^{+0.00008}$	$0.97380_{-0.00008}^{+0.00007}$
Vertical ribbon				
MSE	$(\text{W/cm}^2)^2$	$6.63_{-0.02}^{+0.02}$	$9.61_{-0.02}^{+0.02}$	$9.86_{-0.05}^{+0.06}$
RMSE	W/cm^2	$2.574_{-0.005}^{+0.005}$	$3.100_{-0.003}^{+0.003}$	$3.137_{-0.008}^{+0.009}$
MAE	W/cm^2	$1.995_{-0.003}^{+0.003}$	$2.394_{-0.002}^{+0.002}$	$2.384_{-0.005}^{+0.005}$
MAPE	%	$5.220_{-0.009}^{+0.009}$	$6.140_{-0.008}^{+0.008}$	$6.13_{-0.01}^{+0.01}$
R^2	—	$0.98279_{-0.00007}^{+0.00006}$	$0.97506_{-0.00006}^{+0.00006}$	$0.9744_{-0.0002}^{+0.0001}$

Similarly to what was observed on the large wire dataset analyzed in Section 4.4, models trained in direct visualization tend to perform better than in indirect visualization. In indirect visualization, the validation MSE is from 1.88 to 4.54 times higher than in the direct visualization mode. For comparison, Hobold and da Silva (2019b) showed in their study that the validation and test MSE were approximately 2.5 times higher in indirect visualization compared to direct visualization. This result is closer to this work's error behavior observed for the vertical ribbon dataset.

Tables 14 and 15 show that the heater surface strongly affects model performance.

Table 15 – Training, validation, and test metrics for models trained and evaluated on each heater surface separately in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$.

Metric	Unit	Subset		
		Training	Validation	Test
Large wire				
MSE	$(\text{W/cm}^2)^2$	$9.41_{-0.07}^{+0.06}$	$18.08_{-0.03}^{+0.03}$	$18.14_{-0.04}^{+0.04}$
RMSE	W/cm^2	$3.06_{-0.01}^{+0.01}$	$4.252_{-0.004}^{+0.004}$	$4.258_{-0.005}^{+0.005}$
MAE	W/cm^2	$2.356_{-0.009}^{+0.008}$	$3.247_{-0.003}^{+0.003}$	$3.233_{-0.003}^{+0.003}$
MAPE	%	$5.80_{-0.02}^{+0.02}$	$7.707_{-0.008}^{+0.008}$	$7.69_{-0.01}^{+0.01}$
R^2	—	$0.9761_{-0.0002}^{+0.0002}$	$0.9541_{-0.0001}^{+0.0001}$	$0.9539_{-0.0002}^{+0.0001}$
Small wire				
MSE	$(\text{W/cm}^2)^2$	$22.54_{-0.08}^{+0.08}$	$36.06_{-0.08}^{+0.08}$	$35.47_{-0.04}^{+0.05}$
RMSE	W/cm^2	$4.745_{-0.008}^{+0.008}$	$6.004_{-0.007}^{+0.006}$	$5.956_{-0.004}^{+0.004}$
MAE	W/cm^2	$3.724_{-0.006}^{+0.005}$	$4.655_{-0.002}^{+0.002}$	$4.610_{-0.003}^{+0.003}$
MAPE	%	$10.61_{-0.03}^{+0.03}$	$12.38_{-0.02}^{+0.02}$	$12.10_{-0.01}^{+0.01}$
R^2	—	$0.9592_{-0.0001}^{+0.0001}$	$0.93481_{-0.00010}^{+0.00010}$	$0.93589_{-0.00007}^{+0.00007}$
Horizontal ribbon				
MSE	$(\text{W/cm}^2)^2$	$41.6_{-0.2}^{+0.2}$	$56.1_{-0.1}^{+0.1}$	$52.8_{-0.3}^{+0.2}$
RMSE	W/cm^2	$6.44_{-0.01}^{+0.01}$	$7.49_{-0.01}^{+0.01}$	$7.26_{-0.02}^{+0.02}$
MAE	W/cm^2	$4.790_{-0.009}^{+0.009}$	$5.520_{-0.008}^{+0.008}$	$5.38_{-0.01}^{+0.01}$
MAPE	%	$10.85_{-0.03}^{+0.03}$	$12.38_{-0.03}^{+0.03}$	$11.96_{-0.03}^{+0.03}$
R^2	—	$0.9154_{-0.0003}^{+0.0003}$	$0.8858_{-0.0003}^{+0.0003}$	$0.8924_{-0.0006}^{+0.0006}$
Vertical ribbon				
MSE	$(\text{W/cm}^2)^2$	$19.11_{-0.05}^{+0.05}$	$24.41_{-0.06}^{+0.06}$	$25.48_{-0.08}^{+0.08}$
RMSE	W/cm^2	$4.370_{-0.006}^{+0.006}$	$4.939_{-0.007}^{+0.006}$	$5.046_{-0.008}^{+0.008}$
MAE	W/cm^2	$3.389_{-0.005}^{+0.005}$	$3.824_{-0.006}^{+0.006}$	$3.882_{-0.006}^{+0.007}$
MAPE	%	$8.86_{-0.01}^{+0.01}$	$9.89_{-0.02}^{+0.02}$	$10.09_{-0.02}^{+0.02}$
R^2	—	$0.9504_{-0.0002}^{+0.0002}$	$0.9367_{-0.0002}^{+0.0002}$	$0.9338_{-0.0002}^{+0.0002}$

For instance, the training and validation losses vary nearly 100% when comparing the large and small wire datasets. This is a natural consequence of the different effects of the heater surface on the boiling process, as described in detail in Section 2.1.2.

Table 14 also indicates that, in direct visualization, models achieve the worst performance on the small wire dataset. This can be explained by two factors. The first one is the strong dependency of model performance on the heater surface. The second cause is the low amount of samples in that dataset, as shown in Table 6, in conjunction with the learning curve analysis from Section 5.1.4. Table 6 reports that the small wire

dataset contains approximately 49% of the number of samples of the large wire dataset. Consequently, models trained on the small wire dataset are expected to behave similarly to models trained on 49% of the large wire dataset, corresponding closely to the dataset subsampling rate of 50% depicted in Figure 31. For comparison, at a subsampling rate of 50%, the training, validation, and test losses of models trained on the large wire dataset in direct visualization were $9.46 (\text{W}/\text{cm}^2)^2$, $16.37 (\text{W}/\text{cm}^2)^2$ and $15.51 (\text{W}/\text{cm}^2)^2$, which match very closely the metrics obtained in Table 14 for the small wire dataset.

Finally, compared to the numerical correlations available in the literature and described in Section 2.1.3, the results presented in this study demonstrate the potential of visualization-based machine learning models as a reliable tool for predicting heat flux under different operating conditions. In particular, the error metrics presented in Tables 14 and 15 are comparable to or lower than the uncertainties associated with most numerical correlations found in the literature.

Despite this promising result, the present study trained and evaluated models on each heater surface separately. In order to understand how models generalize to unseen surfaces, Section 5.2.2 presents the results of training and evaluating models on multiple heater surfaces.

5.2.2 Multi-surface evaluation

This Section presents the results of training and evaluating models on multiple surfaces aiming to understand how models behave when trained on images of different heater surfaces and how they generalize to unseen surfaces. This is an extension of the study from Section 5.2.1, which only evaluated models on the dataset they were trained.

In addition to evaluating models on different surfaces, this study also considers *merged* datasets that contain samples from more than one heater surface. Merged datasets are denoted by the merge operator $\bar{\cup}$ and are constructed by proportionally subsampling each component of the merge. For example, the dataset $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$ is constructed by taking 50% of the samples from \mathcal{D}^{LW} and 50% from \mathcal{D}^{SW} . Similarly, the dataset $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}}, \mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$ is constructed by taking 25% of the samples from \mathcal{D}^{LW} , 25% from \mathcal{D}^{SW} , 25% from \mathcal{D}^{HR} , and 25% from \mathcal{D}^{VR} . This proportional subsampling is performed to ensure that the size of the merged datasets is similar to the size of the original datasets^[2]. The sizes of each single-surface and multi-surface dataset are shown in Table 16.

Based on that definition, this study comprises seven datasets: the four experimental datasets \mathcal{D}^{LW} , \mathcal{D}^{SW} , \mathcal{D}^{HR} , and \mathcal{D}^{VR} ; a dataset containing samples from both wires, $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$; a dataset containing samples from both ribbons, $\bar{\cup}(\mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$; and a

^[2] An alternative approach would be to take the union of the datasets, but this would result in a much larger dataset. Its size would be the sum of the components, which would require more computational resources to train and evaluate models. Taking the union of datasets would also provide more information for models to learn than the original datasets, thus making performance comparisons unfair.

Table 16 – Merged datasets split sizes. Each subset (training, validation and test) is a disjoint subsample of the total dataset. The split sizes are shown along with the total dataset size of each dataset. The sizes for \mathcal{D}^{LW} , \mathcal{D}^{SW} , \mathcal{D}^{HR} and \mathcal{D}^{VR} are the same as in Table 6. The size of each merged dataset is the average of the sizes of its component datasets.

Dataset	Training set (70 %)	Validation set (15 %)	Test set (15 %)	Total (100 %)
\mathcal{D}^{LW}	26 416	5661	5662	37 739
\mathcal{D}^{SW}	12 950	2773	2775	18 498
\mathcal{D}^{HR}	58 774	12 593	12 594	83 961
\mathcal{D}^{VR}	51 276	10 987	10 987	73 250
$\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$	19 683	4217	4218	28 118
$\bar{\cup}(\mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$	55 025	11 790	11 790	78 605
$\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}}, \mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$	37 354	8003	8004	53 361

dataset containing samples from all four surfaces, $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}}, \mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$. All datasets were preprocessed by the default preprocessing pipeline.

In order to assess the ability of models to generalize across different surfaces, a model was trained on each dataset and evaluated on the other datasets. Figures 32 and 33 present the results from this study for the direct and indirect visualization modes, respectively. For brevity, only the validation results are presented, but the test results are very similar and lead to the same conclusions.

The results demonstrate that the behavior of models trained and evaluated on different heater surface datasets is similar for both visualization modes. In addition, the main diagonal of the first four rows of Figures 32 and 33 correspond to the results shown in Tables 14 and 15, rounded for convenience.

The most performant models mostly lie on the main diagonal of the heat maps. This indicates that models trained on a specific surface perform best when evaluated on the same surface, which is expected since models tend to perform best on the datasets they were trained. In addition, by inspecting each column of the heat maps, it can be observed that the model that performs best on a given dataset is the one that was trained on it. Moreover, models trained on the merged datasets also perform well when evaluated on the component datasets, meaning that they learn important features from all components. Despite that, models do not generalize to other surfaces without being trained on them, which can be noted from the high error metrics outside the main diagonal and, in the case of merged datasets, when the evaluation is done on non-component datasets.

The contribution of unrelated datasets to the generalization ability of trained models can also be investigated by comparing the model trained on $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$ with the model trained on 50 % of \mathcal{D}^{LW} for the learning curve study. This comparison

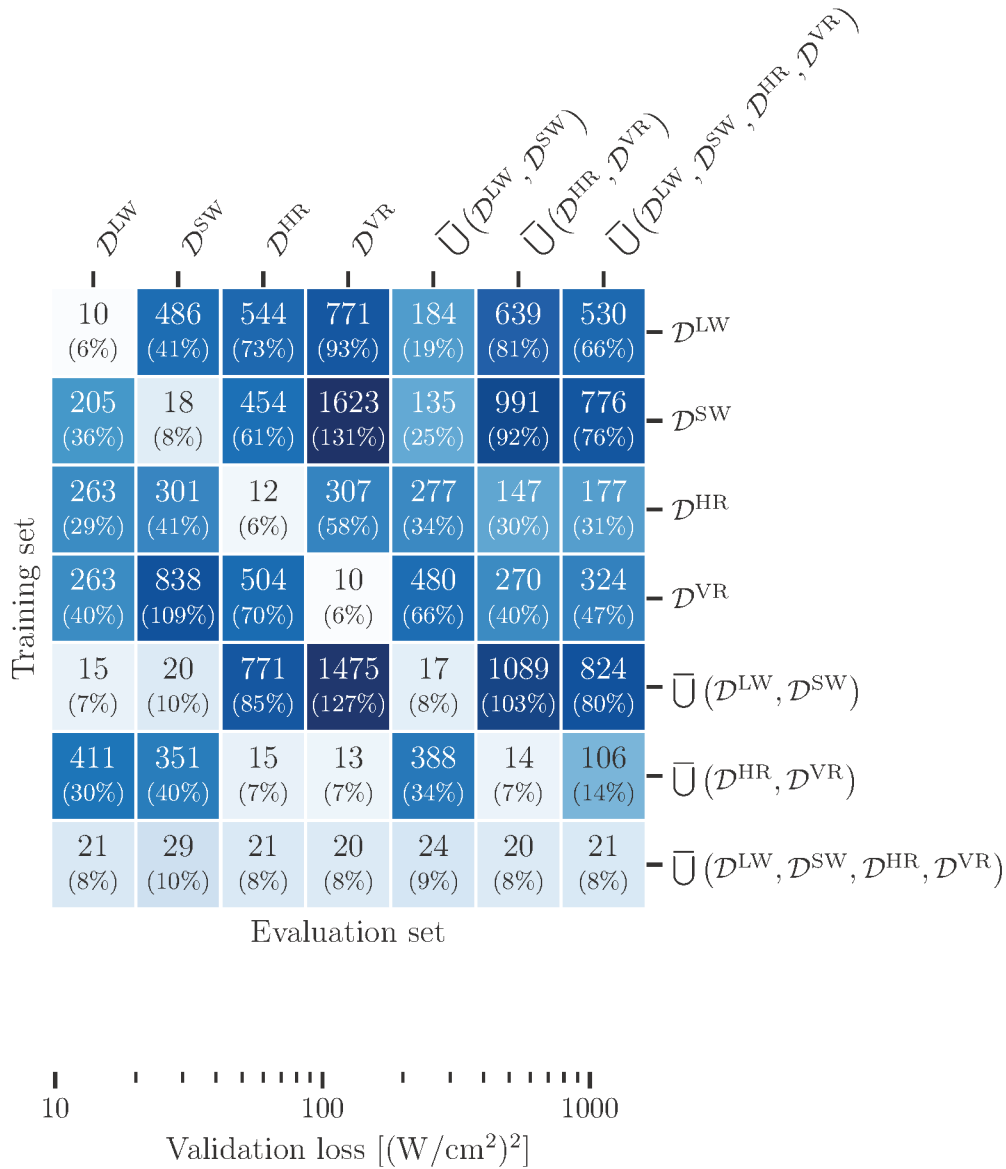


Figure 32 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in direct visualization for $q'' \geq 10 \text{ W/cm}^2$. The vertical axis represents the training set; thus, all results in the same row refer to the same model, trained on the training subset of the dataset denoted at the right of that row. Each column denotes a different evaluation set, denoted at the top of the plot. Each cell shows the validation loss (i.e., the MSE) at its top and the MAPE at its bottom.

can be made because $\bar{\mathcal{U}}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$ also contains 50% of \mathcal{D}^{LW} . By comparing the model trained on $\bar{\mathcal{U}}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$ with the model trained on 50% of \mathcal{D}^{LW} only, it is possible to assess the contribution of the samples from \mathcal{D}^{SW} to the model's generalization ability. In the direct visualization mode, the model trained on $\bar{\mathcal{U}}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}})$ performs approximately 10% better when evaluated on \mathcal{D}^{LW} than the model trained on 50% of \mathcal{D}^{LW} only. Effectively, this implies that the additional 50% of \mathcal{D}^{SW} samples contributed

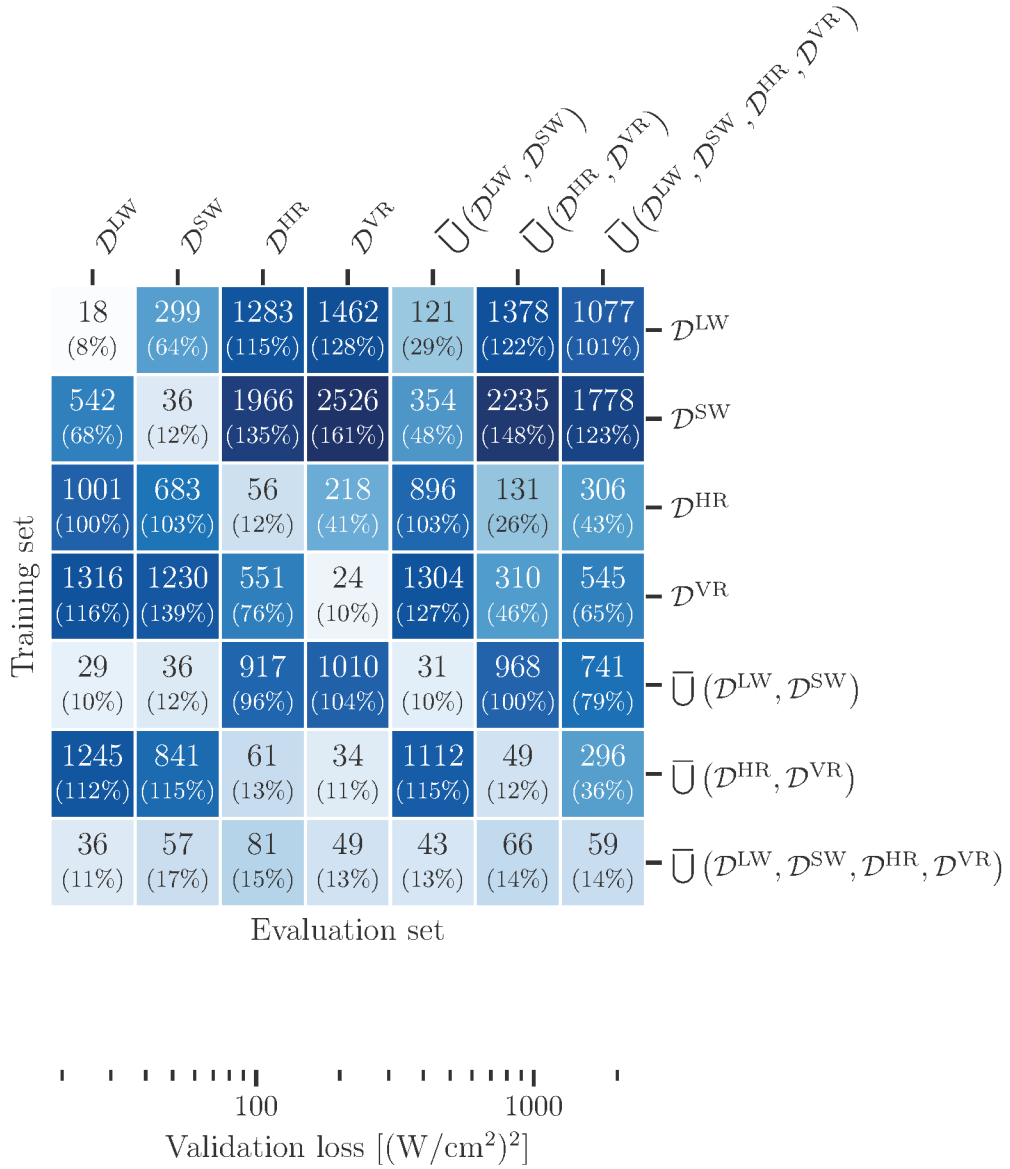


Figure 33 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$. The same notes for Figure 32 are applicable.

to this improvement in performance, meaning that the model generalizes information learned from \mathcal{D}^{SW} to \mathcal{D}^{LW} . In contrast, in the indirect visualization mode, the model trained on $\bar{U}(\mathcal{D}^{LW}, \mathcal{D}^{SW})$ performs approximately 12% worse when evaluated on \mathcal{D}^{LW} than the model trained on 50% of \mathcal{D}^{LW} . This indicates that the trained models cannot generalize information learned from \mathcal{D}^{SW} to \mathcal{D}^{LW} in the indirect visualization mode. A possible explanation for this behavior is the limited amount of information that indirect visualization carries compared to direct visualization. Consequently, more model capacity is required to learn the relationship between frames and the corresponding heat flux in indirect visualization. Since the model architecture, and hence its capacity, is fixed, the model trained on $\bar{U}(\mathcal{D}^{LW}, \mathcal{D}^{SW})$ compromises performance on \mathcal{D}^{LW} to attain reasonable

performance on \mathcal{D}^{SW} .

The models trained on all heater surfaces, $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}}, \mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$, perform relatively well when evaluated on all datasets in both visualization modes. In indirect visualization, the model achieved the worst MAPE of 17% on \mathcal{D}^{SW} , which is still considerably lower than the uncertainties or errors found in the literature. For instance, as mentioned in Section 2.1.3, Scariot (2019) reported errors of approximately 33% on their bi-dimensional pool boiling dataset. In addition, in direct visualization, the model achieved a MAPE as low as 8% when evaluated on $\bar{\cup}(\mathcal{D}^{\text{LW}}, \mathcal{D}^{\text{SW}}, \mathcal{D}^{\text{HR}}, \mathcal{D}^{\text{VR}})$ and 10% in the worst case, when evaluated on \mathcal{D}^{SW} . Those values are considerably close to the MAPE of 7.37% reported by Hobold and da Silva (2019b) and demonstrate that models correctly predict heat flux in pool boiling even in different operating conditions as long as they are trained on them.

In summary, this Section demonstrates that visualization-based deep learning models quantify the heat exchange in pool boiling setups with relatively low associated errors in different operating conditions. In addition, the results consistently show that direct visualization allows models to predict heat flux more accurately than indirect visualization. However, this study also shows that models do not naturally generalize features learned on one surface to other, unseen surfaces. Therefore, it is necessary to train models on all surfaces of interest to achieve good performance on all of them. Exploring domain generalization is left as a suggestion for future work to improve the generalization ability of trained models.

5.3 AUTOMATED MACHINE LEARNING

The final step in the proposed methodology consists of utilizing AutoML algorithms to search for an optimal model architecture and hyperparameters. As explained in Section 2.2.4, AutoML algorithms are efficient search tools that frequently outperform humans in the task, usually finding better models with less computation.

The results from Section 5.2 are promising, but there is still a gap between the most performant model trained in that study and the correlation by Yagov (2009). The optimal model architecture for predicting heat flux in pool boiling is still an open question.

In order to demonstrate the applicability of AutoML, this Section presents the architecture search results in Section 5.3.1. Section 5.3.2 presents the best model found in Section 5.3.1 when evaluated on all heater surfaces.

5.3.1 Architecture search

This Section explores the application of AutoML to search for an optimal model architecture and hyperparameters by executing the AutoML pipeline detailed in Section 4.5. In this study, the AutoML pipeline was executed on the large wire dataset \mathcal{D}^{LW} for both

direct and indirect visualization modes and trained a total of 100 models of different architectures. The greedy AutoML algorithm employed in this study automatically selected the model architecture and training hyperparameters. Table 17 shows all hyperparameters for the best models found by the AutoML pipeline in both direct and indirect visualization modes.

Table 17 – Best hyperparameters found by the AutoML pipeline in both visualization modes.

Hyperparameter	Best value	
	Direct	Indirect
<i>Architecture – Convolutional layers</i>		
Number of convolutional blocks	2	2
Number of consecutive convolutional layers per block	2	2
Convolutional kernel size	3	3
Number of filters in layer #1	128	16
Number of filters in layer #2	64	32
Number of filters in layer #3	32	64
Number of filters in layer #4	32	16
Depthwise separable convolutions?	NO	NO
Apply max-pooling?	NO	NO
Dropout rate	50 %	50 %
<i>Architecture – Reduction layer</i>		
Spatial reduction layer type	Flattening	Flattening
<i>Architecture – Dense layers</i>		
Use batch normalization?	NO	NO
Number of dense layers	2	2
Number of units in layer #1	32	32
Number of units in layer #2	32	16
Dropout rate	0	0
<i>Optimizer</i>		
Optimizer algorithm	Adam	Adam
Learning rate	1×10^{-3}	1×10^{-3}

Table 18 presents the training, validation, and test metrics for the best models found by the AutoML pipeline for both visualization modes. The metrics demonstrate that the proposed AutoML pipeline finds models that perform significantly better than the

baseline. For instance, the best model found by the AutoML pipeline achieved a validation MSE of $4.23 \text{ (W/cm}^2\text{)}^2$ in direct visualization and $6.71 \text{ (W/cm}^2\text{)}^2$ in indirect visualization, representing decreases of approximately 73% and 80%, respectively, compared to the baseline models.

Table 18 – AutoML search results. Metrics are presented only for nucleate and film boiling regimes defined by $q'' \geq 10 \text{ W/cm}^2$. Metrics for the training, validation, and test sets are presented as functions of the visualization mode: direct or indirect.

Metric	Unit	Subset		
		Training	Validation	Test
Direct				
MSE	$\text{(W/cm}^2\text{)}^2$	2.03	4.23	4.42
RMSE	W/cm^2	1.426	2.057	2.102
MAE	W/cm^2	1.07	1.50	1.52
MAPE	%	2.70	3.63	3.67
R^2	—	0.9949	0.9893	0.9888
Indirect				
MSE	$\text{(W/cm}^2\text{)}^2$	1.19	6.71	7.11
RMSE	W/cm^2	1.092	2.591	2.666
MAE	W/cm^2	0.84	1.76	1.77
MAPE	%	2.17	4.13	4.15
R^2	—	0.9970	0.9830	0.9820

In addition to the general improvement compared to the baseline models, the models found via AutoML also reduce the disparity between direct and indirect visualization. Table 11 indicates that the validation loss in indirect visualization is 2.15 times higher than in direct visualization for the baseline models. The best models found by the AutoML pipeline reduce this ratio to 1.59. Models trained in direct visualization are usually expected to perform better than those trained in indirect visualization due to the additional information available in direct visualization that can be utilized for learning and is known to be relevant to the boiling phenomenon. Despite this, the decrease in the difference between the two visualization modes suggests that the best model found by the AutoML pipeline for indirect visualization learned more generalizable features when compared to the baseline.

The results presented in Table 18 also show that the models found via AutoML outperform the correlation by Yagov (2009) in terms of the MAPE. The best model found for direct visualization achieved a MAPE of 3.63%, and the model for indirect visualization achieved a MAPE of 4.13%. For comparison, the correlation by Yagov (2009)

achieved a MAPE of 4.82%. In other words, the AutoML pipeline outperformed the best numerical correlation found in the literature in both visualization modes. To the best of the author's knowledge, this is the first time that visualization-based machine learning models outperform numerical correlations in predicting pool boiling heat transfer.

In addition to the performance gains, the best models found by the AutoML pipeline are also considerably smaller than the baseline models, requiring less RAM and storage capacity and enabling higher prediction throughput. The best model for direct visualization has 850 401 trainable parameters, a fraction of only 3.7% of the 23 041 233 trainable parameters of the baseline model. In indirect visualization, the best model has 3 703 121 trainable parameters, representing 27% of the 13 825 233 trainable parameters of the respective baseline model. This result demonstrates that the AutoML pipeline finds models that are considerably smaller than the baseline ones while outperforming them.

The ability of models to perform better than the baseline while being smaller, and hence having less capacity, might be associated with better connectivity and arrangement of layers. Figures 34 and 35 illustrate the best architectures in direct and indirect visualization, respectively. The most noticeable difference between those architectures and the references illustrated in Figures 10 and 11 is the number and size of the layers. In particular, the best models found by the AutoML pipeline have more convolutional layers and dense blocks than the baseline model but with fewer units. This way, the AutoML pipeline prioritizes deeper models over wider models. As Goodfellow, Bengio, and Courville (2016) explain, deeper models learn more abstract features by composing the learned features from consecutive layers, which might explain the increased depth of the best models found by the AutoML pipeline compared to the baseline models.

Indeed, there is little correlation between model size and performance. Figure 36 presents a scatter plot of the validation loss of the models tried by the AutoML pipeline in direct and indirect visualization versus the number of trainable parameters in the respective models. Note that the baseline model is the largest in the evaluation since its size was set as an upper bound for the search strategy. In both visualization modes, points are well distributed in the plot, meaning that model size does not predict performance.

Figure 36 shows that some points lie on the same vertical line, representing models of the same size with different performance metrics. This is a consequence of the search algorithm that optimized not only the architecture but also other aspects of the training algorithm that do not change the model size, such as the optimizer and its associated learning rate, as well as the dropout ratio.

In addition to providing insight into the behavior of models as a function of their size, the results in Figure 36 also enable design decisions based on trading off performance for compactness. For instance, it is possible to instantiate models more than a hundred times smaller than the baseline model if performance penalties are acceptable in indirect visualization. In this work, model size is a secondary concern, the main goal being to find

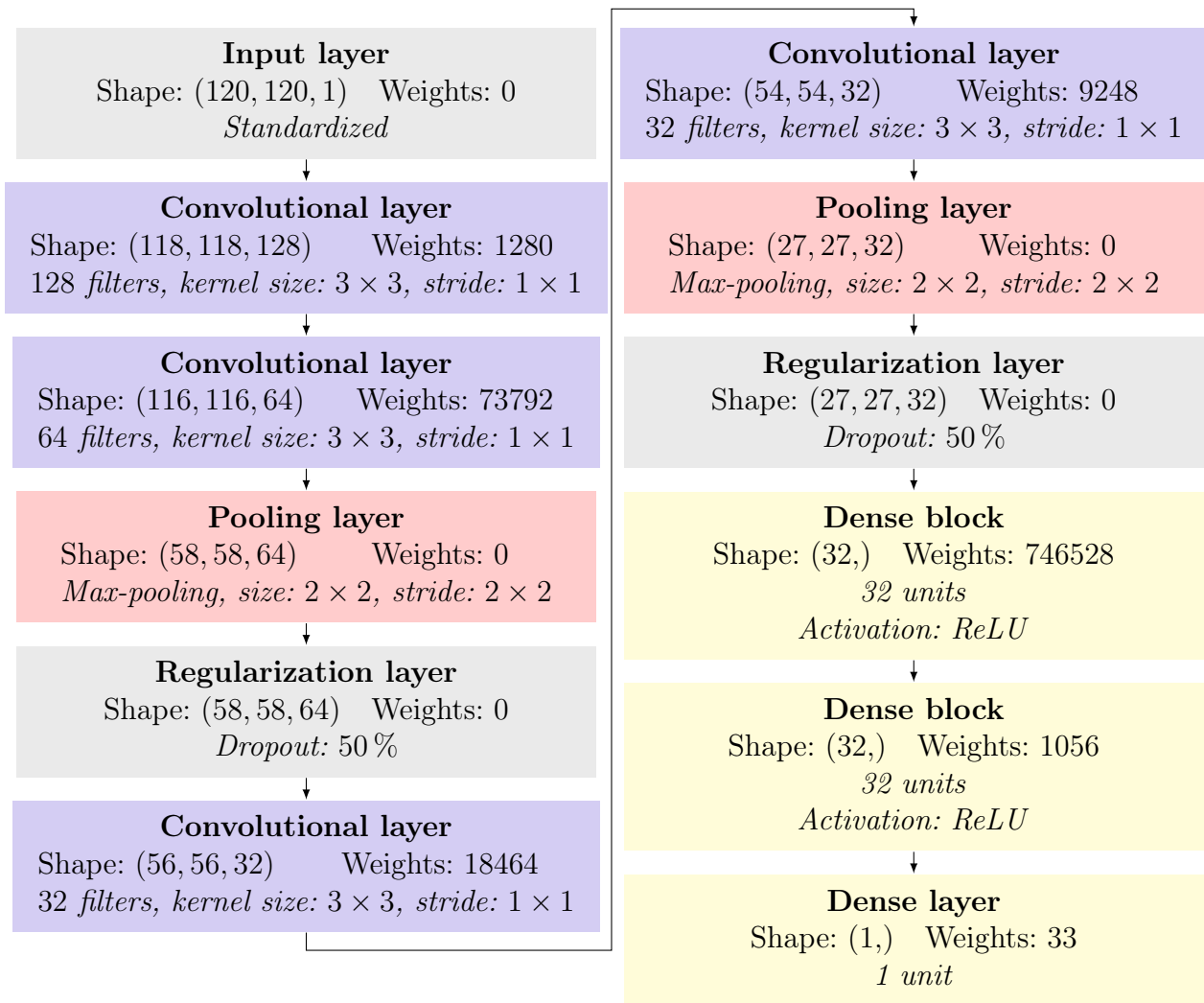


Figure 34 – Diagram of the best model found by the AutoML pipeline in direct visualization. The convolutional blocks are sequences of three layers: a convolutional layer, an activation layer and a max-pooling layer. Similarly, the dense blocks are dense layers followed by activation layers. The input layer is annotated as standardized since image standardization was applied. Each layer is displayed alongside the shape of its outputs and the number of trainable weights it contributes to the model. Note that the weights of dense blocks are arranged in one-dimensional vectors whose shape is denoted as $(\cdot,)$, following the Python notation for tuples of unitary length.

the best model in terms of performance. However, size might be a primary concern in other applications, such as embedded systems. The results in Figure 36 can guide the design of models that are small enough to satisfy throughput or memory footprint constraints.

In summary, the present study shows that AutoML is a viable approach to designing deep learning models for the visualization-based quantification of boiling heat transfer, maximizing performance while reducing model size. The results also show that the AutoML pipeline can find models that outperform the baseline model and numerical correlations in both visualization modes.

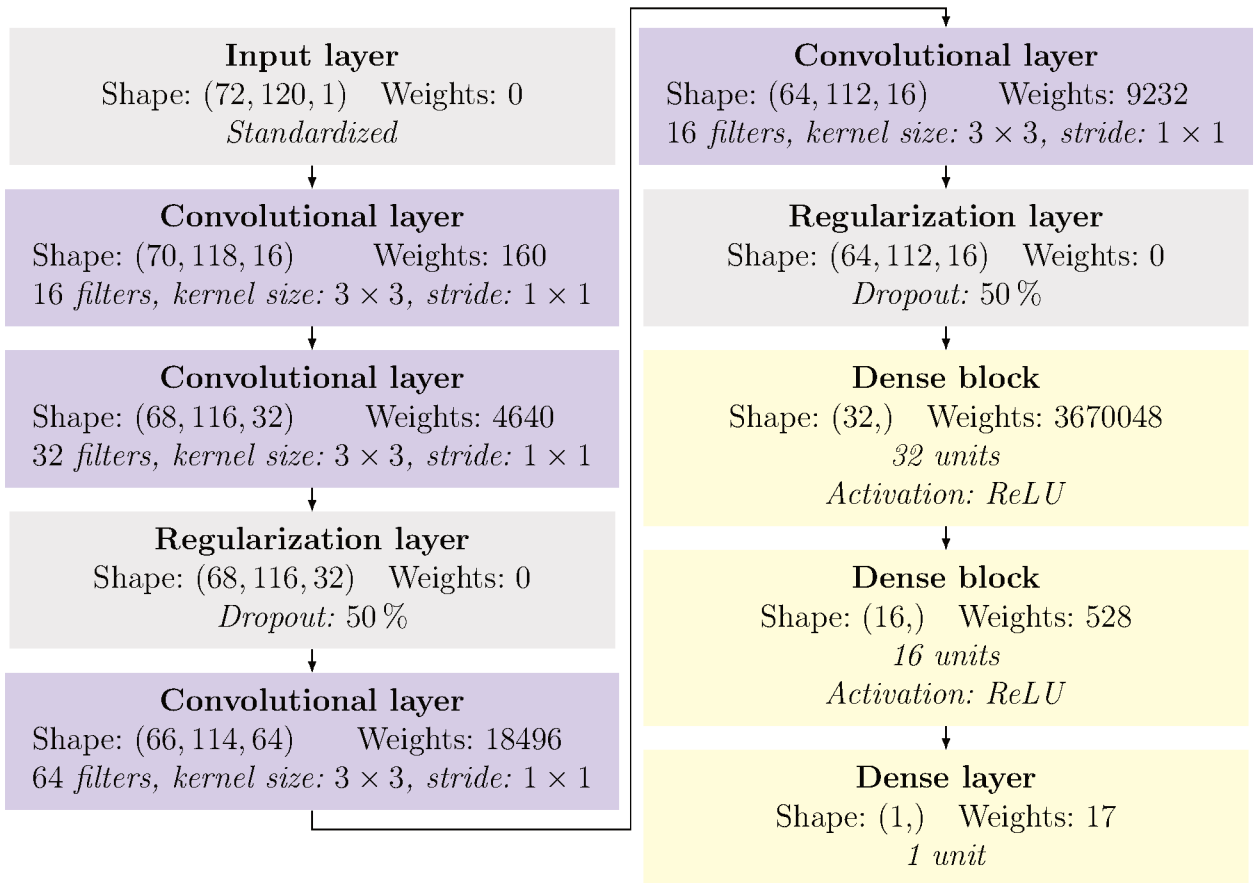


Figure 35 – Diagram of the best model found by the AutoML pipeline in indirect visualization. The same observations for Figure 34 are applicable.

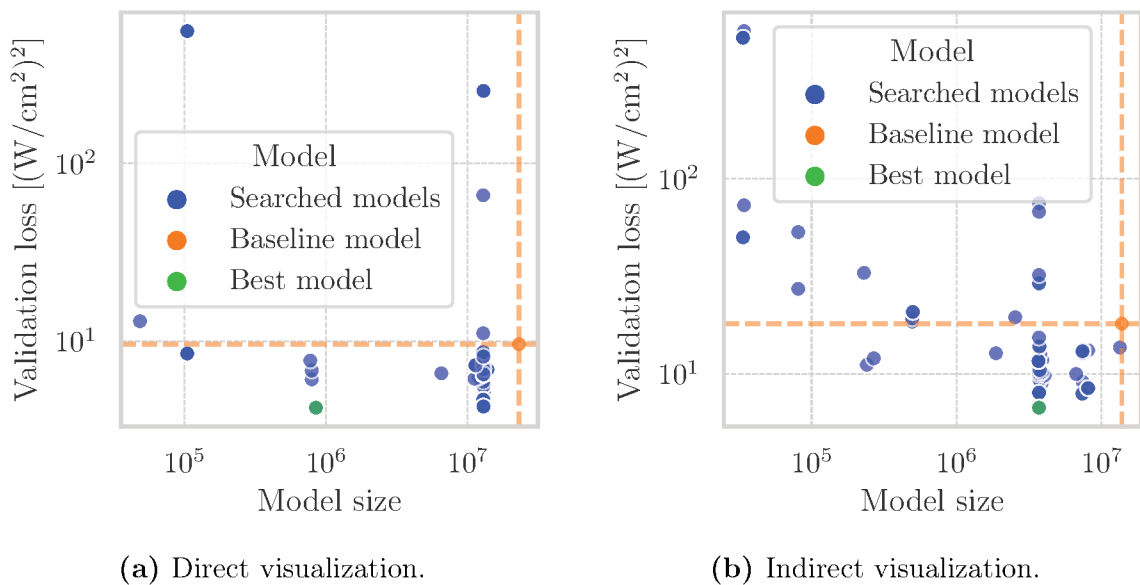


Figure 36 – Validation loss versus model size for both visualization modes.

5.3.2 Multi-surface evaluation

Similarly to Section 5.2.2, this study evaluates the performance of machine learning models when trained and evaluated on multiple heater surfaces aiming to understand their ability to generalize to unseen surfaces.

In this Section, instead of utilizing the baseline model architecture, models are instantiated from the best architectures found by the AutoML pipeline in both visualization modes, depicted in Figures 34 and 35. For each dataset described in Section 5.2.2, the best model architecture is instantiated and retrained on the training subset. Models are retrained utilizing the same optimizer hyperparameters found by the AutoML pipeline. This ensures that not only the model architecture is the best one found but also the training algorithm. The resulting model is then evaluated on the validation subset of all other datasets, and the validation loss (MSE) and MAPE are recorded. Figures 37 and 38 present the results of this process.

In many aspects, the results from this study are similar to those in Section 5.2.2:

1. the results for direct and indirect visualization modes are comparable in behavior, differing mainly in the magnitude of the error metrics. In the cases where models achieved reasonable performance, models trained on direct visualization tend to perform from two to three times better (in terms of MSE) than models trained on indirect visualization;
2. the best performance is found on the main diagonal of the heat maps, which corresponds to models trained and evaluated on the same dataset. This corroborates the hypothesis that models tend to perform better on the datasets they were trained on;
3. models trained on the merged datasets perform well on the component datasets, which means that they learn important features from all components even if multiple surfaces are present in the training set;
4. models do not generalize well to surfaces that do not appear in their training set. In other words, to achieve good performance on a new surface, a model must be trained on a dataset containing samples from that surface.

As a consequence of Items 3 and 4, the model trained on all surfaces is the only one that performs well on all datasets since it is the only model with access to samples from all surfaces during training. This result is also in line with the findings in Section 5.2.2.

The new results from this study demonstrate that the AutoML pipeline proposed in this work can find models that achieve high performance even on datasets containing multiple surfaces as long as the training set contains samples from all surfaces. In the direct visualization mode, the baseline architecture trained and evaluated on $\bar{\cup}(\mathcal{D}^{LW}, \mathcal{D}^{SW}, \mathcal{D}^{HR}, \mathcal{D}^{VR})$ achieved a validation MSE of approximately $21 \text{ (W/cm}^2\text{)}^2$. In contrast, the best architecture found by the AutoML pipeline achieved approximately

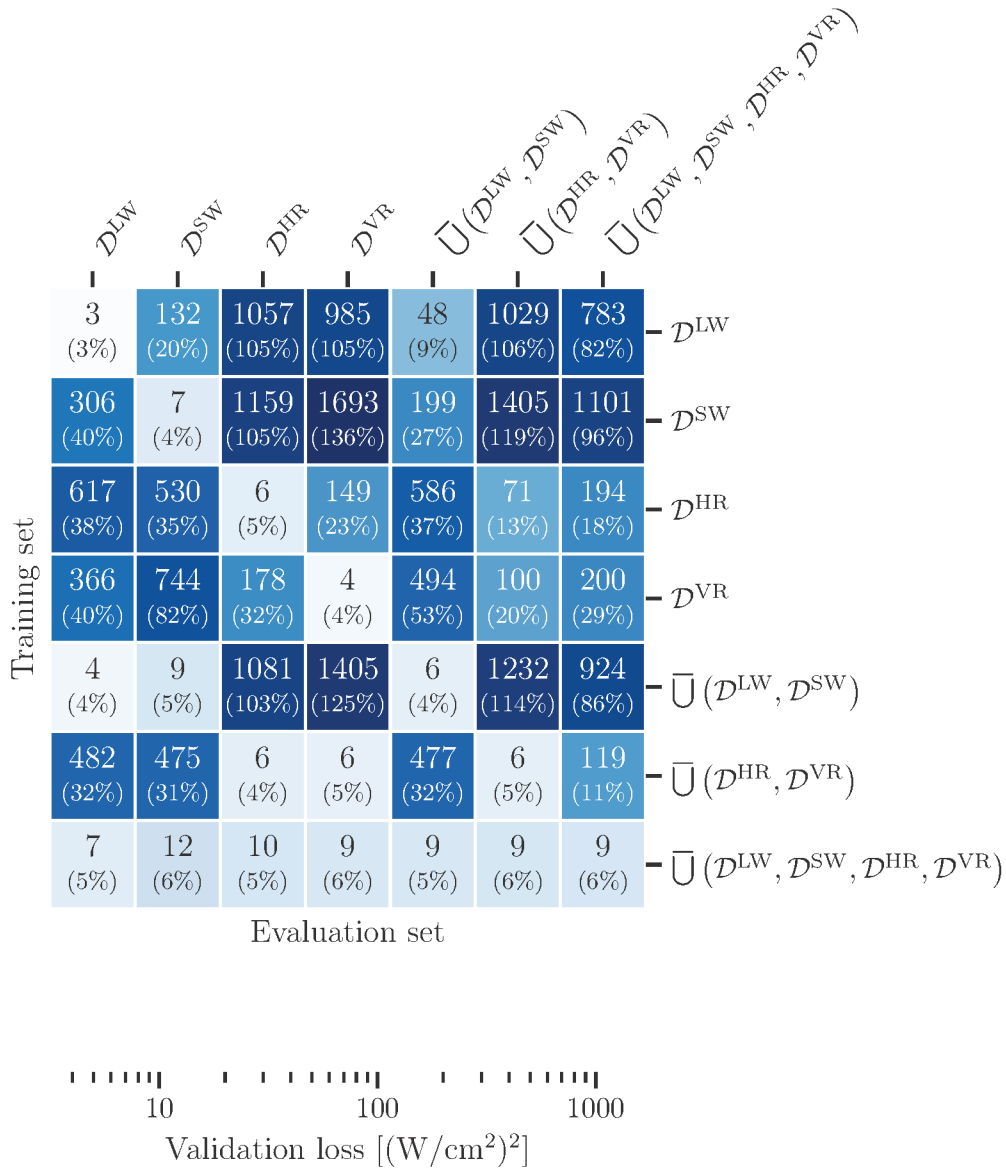


Figure 37 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in direct visualization for $q'' \geq 10 \text{ W/cm}^2$. Models were instantiated from the best architecture found by the AutoML pipeline. The vertical axis represents the training set; thus, all results in the same row refer to the same model, trained on the training subset of the dataset denoted at the right of that row. Each column denotes a different evaluation set, denoted at the top of the plot. Each cell shows the validation loss (i.e., the MSE) at its top and the MAPE at its bottom.

$9 \text{ (W/cm}^2\text{)}^2$, a decrease of more than 57%. In the indirect visualization mode, the validation MSE achieved with the baseline architecture was 59%, whereas the best architecture found by the AutoML pipeline achieved $29 \text{ (W/cm}^2\text{)}^2$, a decrease of almost 51%. Additionally, it can be observed that the best architecture found by the AutoML pipeline, when trained and evaluated on $\bar{U}(\mathcal{D}^{LW}, \mathcal{D}^{SW}, \mathcal{D}^{HR}, \mathcal{D}^{VR})$, achieves similar results to the baseline architecture trained and evaluated only on \mathcal{D}^{LW} in both visualization modes.

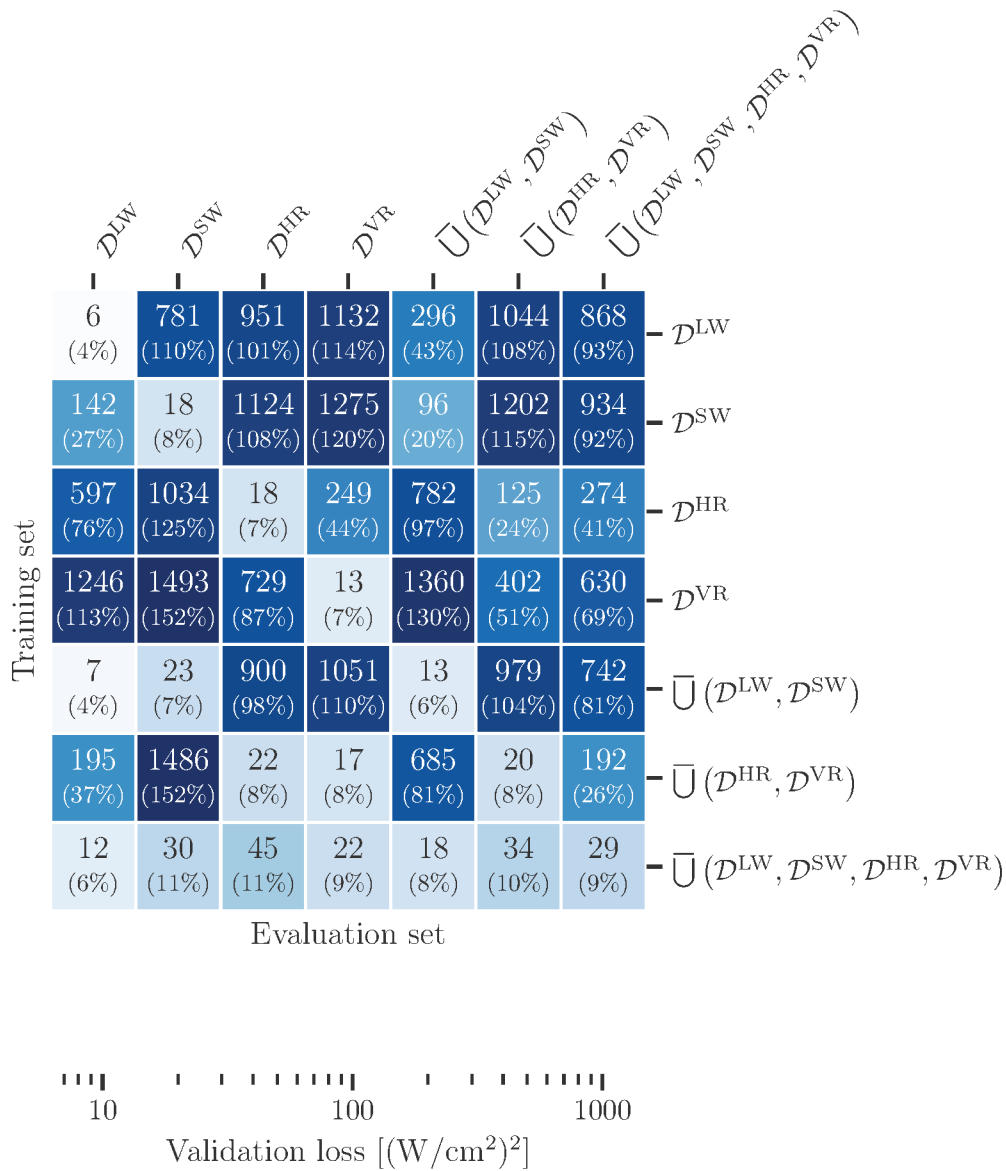


Figure 38 – Heat map plot of the validation metrics for models trained and evaluated on multiple surfaces in indirect visualization for $q'' \geq 10 \text{ W/cm}^2$. The same notes for Figure 37 are applicable.

Finally, compared to the numerical correlations detailed in Section 2.1.3, the proposed methodology stands out as a viable alternative to predict heat flux in a non-intrusive way. In particular, in direct visualization, the AutoML pipeline achieves a validation MAPE of 5.57%, which is only slightly higher than the uncertainty of 4.82% around the correlation by Yagov (2009), and significantly lower than the uncertainties from most correlations that cover multiple heater surfaces.

6 CONCLUSION

This Thesis demonstrated the applicability of machine learning models to estimate the heat flux dissipated from different heater surfaces in pool boiling setups. This closing Chapter summarizes the findings of this work, highlighting the most important contributions to the literature and the limitations of the proposed methodology. For future work, Section 6.2 provides suggestions and recommendations based on the discoveries and limitations of this Thesis.

6.1 SUMMARY

Chapter 5 presented and discussed the results obtained in this Thesis. First, Section 5.1 presented the data preprocessing pipeline optimization results that justify the design decisions made in Section 4.2:

1. image standardization reduces the validation error by nearly 39 % in direct visualization and 47 % in indirect visualization;
2. image downscaling helps to improve model performance. However, the optimal downscaling factor is dependent on the visualization mode. For direct visualization, the optimal downscaling factor is $f_{ds} = 6$, while for indirect visualization, the optimal downscaling factor is $f_{ds} = 5$. Image downscaling reduced the validation error by 59–74 %, depending on the downscaling factor and the visualization mode;
3. a visualization window width of 60 % of the original frame width is optimal for both direct and indirect visualization;
4. the learning curve study shows that the baseline model requires 100 % of the baseline dataset for maximum performance since dataset subsampling increases the validation error. However, it was also observed that the validation error stabilized after 80 % of the baseline dataset was used for training, suggesting that no additional data is required to achieve maximum performance;

Subsequently, Section 5.2 trained and evaluated models on datasets containing different heater surfaces to assess their generalization capabilities. The results showed that:

1. models perform best in direct visualization, suggesting that they make use of the bubble dynamics before departure to estimate the heat flux;
2. models only performed well on the heater surfaces contained in their training set but did not generalize well to unseen surfaces;
3. models trained on merged datasets (that is, datasets containing samples from multiple heater surfaces) performed well on the component datasets, which means that they learn meaningful features from the different heater surfaces in their training set;

4. the model trained on samples from all heater surfaces performed well on all datasets, achieving errors comparable to the baseline model and outperforming most numerical correlations and the model by Scariot (2019);

Finally, Section 5.3 presented the results obtained by the AutoML pipeline on the baseline dataset and the datasets containing multiple heater surfaces:

1. the AutoML pipeline found model architectures that performed significantly better than the baseline model. The best models decreased the validation error by 73 % in direct visualization and 80 % in indirect visualization;
2. the best models found via AutoML outperformed the best numerical correlation found in the literature by Yagov (2009) on the baseline dataset in both visualization modes. This is the first time that a visualization-based machine learning model outperforms numerical correlations in the problem of estimating heat flux in pool boiling setups;
3. when trained and evaluated on the dataset containing all heater surfaces, the best models found via AutoML achieved a similar error to the baseline model trained and evaluated only on a single surface. The calculated MAPE was 5.57 %, which is slightly higher than the uncertainty around the correlation by Yagov (2009) and significantly lower than most numerical correlations considered;
4. the best models found via AutoML were also significantly smaller than the baseline models. The best model for direct visualization had only 3.7 % of the parameters of the baseline model. The best model for indirect visualization had 27 % of the number of parameters of the baseline model for that visualization mode. This means that AutoML finds better models with fewer parameters, which is crucial for real-time applications both in terms of time-to-inference requirements and memory and storage constraints;

This summary demonstrates that the main objective of this work, stated in Section 1.1, was achieved along with the associated specific objectives. The results obtained in this work show that CNNs can estimate the heat flux dissipated from different heater surfaces in pool boiling setups. However, models do not generalize well to unseen surfaces and hence need to be trained on datasets containing samples from all heater surfaces of interest. The results also show that AutoML is a powerful tool for automatically searching for better model architectures with minimal human intervention.

6.2 RECOMMENDATIONS AND SUGGESTIONS FOR FUTURE STUDIES

Based on the findings presented in Chapter 5 and the aspects left unexplored in this work, this Section presents several recommendations for future studies to follow to improve their results and suggests several topics that could be further investigated in future studies.

The main recommendations derived from this Thesis can be enumerated as:

1. **image standardization:** image standardization significantly reduces the validation error without incurring significant computational costs. Note that image standardization is different from image normalization, which was applied to the reference dataset;
2. **downscaling and decreasing the visualization window width:** image downscaling and decreasing the visualization window width were shown to significantly reduce the validation error while reducing the dimensionality of datasets. However, the values found in this study are likely dependent on the experimental setup and the heater surfaces used. Therefore, it is recommended to perform these studies for each dataset to determine the optimal values;
3. **learning curve analysis:** performing a learning curve analysis as soon as a complete dataset is available allows the determination of the optimal number of samples to use for training in order to decide if more data is required or if datasets can be subsampled to accelerate training;
4. **automated machine learning:** if enough computational resources are available, utilize AutoML to search for better model architectures, which might improve the performance of models significantly while reducing their size;

Regarding the experimental setup described in Chapter 3, the following suggestions can be implemented to allow further analyses:

1. **control the power supply through a computer:** with this, it would be possible to programmatically impose the thermal power q dissipated by the test sample, allowing for automated experimental runs and finer control of the heat flux levels. In order to do that, a driver officially distributed by National Instruments must be installed, but it was unavailable by the time the experiments were run in this work;
2. **control the auxiliary heaters with a relay:** with this, it is possible to implement an on/off control subsystem to keep the liquid temperature approximately constant at any level. This would make it feasible to study subcooled boiling;
3. **implement a test sample temperature measurement subsystem:** the measurement of the test sample temperature would allow the calculation of the heat transfer coefficient, which is commonly the variable to be maximized to optimize thermal systems for efficiency. The heat transfer coefficient is also the most common output from numerical correlations. A failed attempt to implement it was made in this work, as described in Section 3.6;

Finally, the following aspects related to the machine learning pipeline were left unexplored in this work and could be investigated in future studies:

1. **data augmentation:** data augmentation consists of artificially generating new dataset samples from existing ones. This can be done by applying random transformations to the images, such as rotations, translations, scaling, and cropping. Data augmentation might reduce the performance of models on the training set since it increases the complexity of the problem, but it can help models generalize better to unseen data;
2. **transfer learning:** transfer learning consists of using a model trained on a different problem to initialize the weights of a new model. This can be done by freezing the weights of the first layers of the model and training only the last layers. Another strategy is to fine-tune the weights of all model layers using a smaller dataset. Specifically, in the problem of estimating heat flux on different heater surfaces in pool boiling setups, a baseline model could be trained on the baseline dataset and only fine-tuned on the other datasets containing different heater surfaces. This would potentially allow the use of a smaller dataset for fine-tuning, which would speed up training;
3. **quantization-aware training:** quantization-aware training consists of training a model with quantized weights, parameters of reduced precision types such as `float16`. This significantly reduces the number of bits required to represent the weights and activations of the model, thus reducing the time-to-inference and the overall computational cost, including the RAM and storage footprints. However, it might also reduce the performance of the model. According to Abadi et al. (2015b), quantization-aware training can reduce the total model size four times and improve latency from 1.5–4 times while keeping model performance almost unchanged;
4. **search strategies:** in this work, the selected AutoML search strategy was the *greedy* algorithm. Initial, informal tests showed that this algorithm could find better models with fewer trials compared to alternatives such as the Bayesian or the hyperband algorithms. However, these tests were not rigorously conducted. Hence, a more systematic analysis of the available search strategies is suggested. A better search strategy might require fewer trials to find more performant and smaller models;
5. **pre-trained model fine-tuning:** in Section 4.5, it was mentioned that the AutoML pipeline employed in this work does not include pre-trained models such as ResNet (HE et al., 2015) or Xception (CHOLLET, 2016). Future studies can address this limitation of the pipeline. Fine-tuning publicly available pre-trained models can significantly reduce the number of trials required to find better models and increase the overall performance of the pipeline;

REFERENCES

- ABADI, Martín et al. **Metrics Overview**. [S.l.: s.n.], 2015. Available from: https://www.tensorflow.org/api_docs/python/tf/keras/metrics.
- ABADI, Martín et al. **Quantization aware training**. [S.l.: s.n.], 2015. Available from: https://www.tensorflow.org/model_optimization/guide/quantization/training.
- ABADI, Martín et al. **R2Score**. [S.l.: s.n.], 2015. Available from: https://www.tensorflow.org/api_docs/python/tf/keras/metrics/R2Score.
- ABADI, Martín et al. **TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems**. [S.l.: s.n.], 2015. Available from: <https://www.tensorflow.org/>.
- BEJAN, A.; KRAUS, A.D. **Heat Transfer Handbook**. [S.l.]: Wiley, 2003. (Heat Transfer Handbook, v. 1). ISBN 9780471390152.
- BERGMAN, T.L.; INCROPERA, F.P.; DEWITT, D.P.; LAVINE, A.S. **Fundamentals of Heat and Mass Transfer**. [S.l.]: Wiley, 2011. ISBN 9780470501979.
- BERGSTRA, James; BARDENET, Rémi; BENGIO, Yoshua; KÉGL, Balázs. Algorithms for Hyper-Parameter Optimization. In: SHAWE-TAYLOR, J.; ZEMEL, R.; BARTLETT, P.; PEREIRA, F.; WEINBERGER, K.Q. (Eds.). **Advances in Neural Information Processing Systems**. [S.l.]: Curran Associates, Inc., 2011. v. 24.
- BISHOP, Christopher M. **Pattern recognition and machine learning**. [S.l.]: springer, 2006.
- CAMPAGNOLA, Luke. **PyQtGraph**. [S.l.]: GitHub, 2019. Available from: <https://github.com/pyqtgraph/pyqtgraph>. Visited on: 12 Nov. 2022.
- ÇENGEL, Y. A.; GHAJAR, A. J. **Transferência de Calor e Massa**. New York: McGraw Hill Brasil, 2009. ISBN 9788580551280.
- CHOLLET, F. et al. **Keras**. [S.l.: s.n.], 2015. <https://keras.io>. Visited on: 29 Dec. 2022.

- CHOLLET, F. Xception: Deep Learning with Depthwise Separable Convolutions. **CoRR**, abs/1610.02357, 2016. arXiv: 1610.02357.
- CHOLLETT, F. **Deep Learning with Python**. [S.l.]: Manning, 2018. ISBN 9781617294433.
- COMELLI, Ruan C. **Boiling Learning**. [S.l.]: GitHub, 2023. <https://github.com/ruancomelli/boiling-learning>.
- EMIR, Tolga; OURABI, Hamza; BUDAKLI, Mete; ARIK, Mehmet. Parametric Effects on Pool Boiling Heat Transfer and Critical Heat Flux: A Critical Review. **Journal of Electronic Packaging**, v. 144, n. 4, Apr. 2022. 040801. ISSN 1043-7398.
- FRIEDMAN, Jerome; HASTIE, Trevor; TIBSHIRANI, Robert. **The elements of statistical learning: Data Mining, Inference, and Prediction**. 2. ed. [S.l.]: Springer series in statistics New York, 2001. (Springer Series in Statistics).
- GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. [S.l.]: MIT press, 2016.
- GORENFLO, Dieter; BAUMHÖGGER, Elmar; HERRES, Gerhard; KOTTHOFF, Stephan. Prediction methods for pool boiling heat transfer: A state-of-the-art review. **International Journal of Refrigeration**, v. 43, p. 203–226, 2014. ISSN 0140-7007.
- GORENFLO, Dieter; KENNING, David. Pool Boiling. In: STEPHAN, Peter; KABELAC, Stephan; KIND, Matthias; MARTIN, Holger; MEWES, Dieter; SCHABER, Karlheinz (Eds.). **VDI Heat Atlas**. 2. ed. Berlin: Springer Berlin Heidelberg, 2010. (VDI-Buch). H2, p. 757–792. ISBN 9783540778769.
- HARRIS, Charles R. et al. Array programming with NumPy. **Nature**, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, Sept. 2020.
- HASHEMIAN, H.M.; JIANG, Jin. Nuclear plant temperature instrumentation. **Nuclear Engineering and Design - NUCL ENG DES**, v. 239, p. 3132–3141, Dec. 2009.
- HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. **Deep Residual Learning for Image Recognition**. [S.l.: s.n.], 2015. arXiv: 1512.03385 [cs.CV].

HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep residual learning for image recognition. In: PROCEEDINGS of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. P. 770–778.

HOBOLD, Gustavo M.; DA SILVA, Alexandre K. Analysis of neural network architecture for pool boiling regime identification. In: 10TH International Conference on Boiling & Condensation Heat Transfer. Nagasaki, Japan: [s.n.], 2018.

HOBOLD, Gustavo M.; DA SILVA, Alexandre K. Automatic detection of the onset of film boiling using convolutional neural networks and Bayesian statistics. **International Journal of Heat and Mass Transfer**, v. 134, p. 262–270, 2019. ISSN 0017-9310.

HOBOLD, Gustavo M.; DA SILVA, Alexandre K. Machine learning classification of boiling regimes with low speed, direct and indirect visualization. **International Journal of Heat and Mass Transfer**, v. 125, p. 1296–1309, 2018. ISSN 0017-9310.

HOBOLD, Gustavo M.; DA SILVA, Alexandre K. Visualization-based nucleate boiling heat flux quantification using machine learning. **International Journal of Heat and Mass Transfer**, v. 134, p. 511–520, 2019. ISSN 0017-9310.

HOFMANN, Heike; KAFADAR, Karen; WICKHAM, Hadley. **Letter-value plots: Boxplots for large data**. [S.l.], 2011.

HUGHES, Matthew T.; KINI, Girish; GARIMELLA, Srinivas. Status, Challenges, and Potential for Machine Learning in Understanding and Applying Heat Transfer Phenomena. **Journal of Heat Transfer**, v. 143, n. 12, Oct. 2021. 120802. ISSN 0022-1481.

HUNTER, J. D. Matplotlib: A 2D graphics environment. **Computing in Science & Engineering**, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.

HUTTER, Frank; HOOS, Holger H.; LEYTON-BROWN, Kevin. Sequential Model-Based Optimization for General Algorithm Configuration. In: COELLO, Carlos A. Coello (Ed.). **Learning and Intelligent Optimization**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. P. 507–523.

IAEA. **Modern Instrumentation and Control for Nuclear Power Plants: A Guidebook**. Viena, 1999. (Technical Reports Series, 387). Available from: <https://www-pub.iaea.org/MTCD/publications/>. Visited on: 26 Oct. 2019.

- JAMES, Gareth; WITTEN, Daniela; HASTIE, Trevor; TIBSHIRANI, Robert. **An introduction to statistical learning**. [S.l.]: Springer, 2013. v. 112.
- JCGM. **Evaluation of measurement data**: Guide to the expression of uncertainty in measurement. Version 1. [S.l.], 2008.
- JIN, Haifeng; SONG, Qingquan; HU, Xia. Auto-Keras: An Efficient Neural Architecture Search System. In: ACM. PROCEEDINGS of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. [S.l.: s.n.], 2019. P. 1946–1956.
- JUDD, Ross L.; HWANG, Ki Soon. A Comprehensive Model for Nucleate Pool Boiling Heat Transfer Including Microlayer Evaporation. **Journal of Heat Transfer-Transactions of the ASME**, v. 98, p. 623–629, 1976.
- JUNG, S.; KIM, H. Effects of surface orientation on nucleate boiling heat transfer in a pool of water under atmospheric pressure. **Nuclear Engineering and Design**, v. 305, p. 347–358, 2016. ISSN 0029-5493.
- KANDLIKAR, S.G. **Handbook of Phase Change: Boiling and Condensation**. [S.l.]: Taylor & Francis, 1999. ISBN 9781560326342.
- KIM, J. H.; YOU, S. M.; PAK, J. Y. Effects of heater size and working fluids on nucleate boiling heat transfer. **International Journal of Heat and Mass Transfer**, v. 49, n. 1, p. 122–131, 2006. ISSN 0017-9310.
- KIM, M.; KIM, S. J. A mechanistic model for nucleate pool boiling including the effect of bubble coalescence on area fractions. **International Journal of Heat and Mass Transfer**, v. 163, p. 120453, 2020. ISSN 0017-9310.
- KINGMA, Diederik P.; BA, Jimmy. **Adam: A Method for Stochastic Optimization**. [S.l.]: arXiv, 2014. Available from: <https://arxiv.org/abs/1412.6980>.
- KOIZUMI, Yasuo; SHOJI, Masahiro; MONDE, Masanori; TAKATA, Yasuyuki; NAGAI, Niro. **Boiling: Research and Advances**. 1. ed. [S.l.]: Elsevier, 2017.
- KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey. ImageNet Classification with Deep Convolutional Neural Networks. **Neural Information Processing Systems**, v. 25, Jan. 2012.

LI, Lisha; JAMIESON, Kevin; DESALVO, Giulia; ROSTAMIZADEH, Afshin; TALWALKAR, Ameet. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. arXiv, 2016.

LOSHCHILOV, Ilya; HUTTER, Frank. **Decoupled Weight Decay Regularization**. [S.l.: s.n.], 2019. arXiv: 1711.05101 [cs.LG].

MIKIĆ, B. B.; ROHSENOW, W. M. A New Correlation of Pool-Boiling Data Including the Effect of Heating Surface Characteristics. **Journal of Heat Transfer-transactions of The Asme**, v. 91, p. 245–250, 1969.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997. (Springer Series in Statistics). ISBN 0070428077.

MODIN PROJECT. **modin**. Version 0.17.0. 2018. Available from: <https://github.com/modin-project/modin>. Visited on: 12 Nov. 2022.

MURPHY, Kevin P. **Machine Learning: A Probabilistic Perspective**. Cambridge: The MIT Press, 2012. ISBN 978-0-262-01802-9.

NATIONAL INSTRUMENTS. **nidaqmx-python**. Version 0.5.7. 2017. Available from: <https://github.com/ni/nidaqmx-python>. Visited on: 12 Oct. 2019.

NEUTELINGS, Izaak. **CodeSnippets**. [S.l.]: GitHub, 2022. Available from: <https://github.com/IzaakWN/CodeSnippets>. Visited on: 10 Jan. 2023.

NICKOLLS, John; BUCK, Ian; GARLAND, Michael; SKADRON, Kevin. Scalable Parallel Programming with CUDA. **Queue**, v. 6, p. 40–53, Mar. 2008.

NUKIYAMA, Shiro. The maximum and minimum values of the heat Q transmitted from metal to boiling water under atmospheric pressure. **International Journal of Heat and Mass Transfer**, v. 9, n. 12, p. 1419–1433, 1966. Translated from Journal of the Society of Mechanical Engineers, 37, 367–374, Japan, 1934. ISSN 0017-9310.

NVIDIA. **DEEP LEARNING PERFORMANCE DOCUMENTATION**. [S.l.], 2022. Available from: <https://docs.nvidia.com/deeplearning/performance/mixed-precision-training/index.html#opt-tensor-cores>. Visited on: 2 Dec. 2022.

O'MALLEY, T.; BURSZTEIN, E.; LONG, J.; CHOLLET, F.; JIN, H.; INVERNIZZI, L., et al. **KerasTuner**. [S.l.: s.n.], 2019. <https://github.com/keras-team/keras-tuner>.

OMEGA ENGINEERING. **1400°F Max 80 Nickel 20 Chromium Resistance Heating Ribbon**: NCCR-17-100 Model Options. 2022. Available from: <https://www.omega.com/en-us/accessories/electrical-components/heating-wire-and-cables/ncrr-series/p/NCCR-17-100>. Visited on: 28 Dec. 2022.

OMEGA ENGINEERING. **2100°F Max 80 Nickel 20 Chromium Resistance Heating Wire**: NI80-020-200 Model Options. 2022. Available from: <https://www.omega.com/en-us/accessories/electrical-components/heating-wire-and-cables/ni80/p/NI80-020-200>. Visited on: 28 Dec. 2022.

OMEGA ENGINEERING. **2100°F Max 80 Nickel 20 Chromium Resistance Heating Wire**: NI80-010-200 Model Options. 2022. Available from: <https://www.omega.com/en-us/accessories/electrical-components/heating-wire-and-cables/ni80/p/NI80-010-200>. Visited on: 28 Dec. 2022.

PAGE, Jimmy; PLANT, Robert. **Led Zeppelin IV: Stairway to Heaven**. New York City, US: Atlantic Records, 1971. Led Zeppelin.

PYTHON CORE TEAM. **Python: A dynamic, open source programming language**. [S.l.], 2015. Python 3.10.8. Available from: <https://www.python.org/>. Visited on: 27 Nov. 2022.

ROHSENOW, W. M. A Method of Correlating Heat-Transfer Data for Surface Boiling of Liquidss. **Transactions of the American Society of Mechanical Engineers**, v. 74, n. 6, p. 969–975, 1952.

ROHSENOW, W. M.; HARTNETT, J. P.; CHO, Y. I. **Handbook of Heat Transfer**. 3. ed. [S.l.]: McGraw-Hill, 1998.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: THE IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], June 2018.

SCARIOT, Vinicius Kramer. **APPLICATION OF STATISTICALLY-BASED METHODS TO HEAT TRANSFER PROBLEMS**. 2019. S. 148. Mestrado em Engenharia Mecânica – Universidade Federal de Santa Catarina, Florianópolis.

- SNOEK, Jasper; LAROCHELLE, Hugo; ADAMS, Ryan P. **Practical Bayesian Optimization of Machine Learning Algorithms**. [S.l.]: arXiv, 2012. Available from: <https://arxiv.org/abs/1206.2944>.
- SONG, Qingquan; JIN, Haifeng; HU, Xia. **Automated Machine Learning in Action**. 1. ed. [S.l.]: Manning Publications, 2022. ISBN 1617298050; 9781617298059.
- SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. **Journal of Machine Learning Research**, v. 15, n. 56, p. 1929–1958, 2014.
- TOFALLIS, Chris. A Better Measure of Relative Prediction Accuracy for Model Selection and Model Estimation. **Journal of the Operational Research Society**, n. 66, p. 1352–1362, 2015.
- VIRTANEN, Pauli et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. **Nature Methods**, v. 17, p. 261–272, 2020.
- WALT, Stéfan van der; SCHÖNBERGER, Johannes L.; NUNEZ-IGLESIAS, Juan; BOULOGNE, François; WARNER, Joshua D.; YAGER, Neil; GOUILLART, Emmanuelle; YU, Tony; CONTRIBUTORS, the scikit-image. scikit-image: image processing in Python. **PeerJ**, v. 2, e453, June 2014. ISSN 2167-8359.
- WANG, Zhou; BOVIK, A.C.; SHEIKH, H.R.; SIMONCELLI, E.P. Image quality assessment: from error visibility to structural similarity. **IEEE Transactions on Image Processing**, v. 13, n. 4, p. 600–612, 2004.
- WASKOM, Michael L. seaborn: statistical data visualization. **Journal of Open Source Software**, The Open Journal, v. 6, n. 60, p. 3021, 2021.
- YAGOV, V. V. Nucleate boiling heat transfer: possibilities and limitations of theoretical analysis. **Heat and Mass Transfer**, v. 45, p. 881–892, 2009.

APPENDIX A – DATA SHEETS

A.1 NI80-020-200 AND NI80-010-200 (OMEGA ENGINEERING)

Resistance Heating Wire

Nickel-Chromium Alloy

80% Nickel/20% Chromium

- ✓ Withstands High Temperatures up to 1150°C (2100°F)
- ✓ Quick Heating, Long Life
- ✓ Corrosion Resistant
- ✓ Used to Make Straight or Helical Coil Resistance Heaters
- ✓ Convenient 15 m (50') and 60 m (200') Spools Available

OMEGA™ NIC80 wire is a resistance heating wire comprised of 80% Nickel and 20% Chromium. NIC80 wire is commonly used as a resistor at elevated temperatures. NI/CR-80/20 is essential for resistor elements in high temperature applications such as electric furnaces, electric ranges and radiant heaters operating at temperatures up to 1150°C (2100°F).

In addition to these qualities and standard uses, it has found wide application in technical applications due to its combination of high electrical resistance and its temperature coefficient of resistance much less than that of Nickel-Chrome 60.



Specifications

Composition: 80% Ni, 20% Cr
Specific Resistance: 650 Ω per circular mil-foot at 20°C (68°F). See table below for multiplication factors to obtain resistance at other temperatures.
Specific Gravity: 8.41
Density: 0.304 lb/in³

Melting Point: Approx 1400°C (2550°F)
Nominal Coefficient of Linear Expansion: 0.000017 (10 to 1000°C)
Tensile Strength (lb/in²) at 20°C (68°F):
Soft Annealed: 100,000
Nominal Temperature Coefficient of Resistance: 0.00011 Ω/Ω/°C (20 to 500°C)

Factor by Which Resistance at Room Temperature Is to Be Multiplied to Obtain Resistance at Indicated Temperatures (These figures are given as a basis for engineering calculations and represent average material as supplied.)											
Temp °C	20	93	204	315	427	538	649	760	871	982	1093°C
Temp °F	68	200	400	600	800	1000	1200	1400	1600	1800	2000°F
Factor	1.000	1.016	1.037	1.054	1.066	1.070	1.064	1.062	1.066	1.072	1.078

To Order										
AWG	Dia. mm (1")	Ω per ft @ 20°C (68°F)	Current Temperature Characteristics* °C (°F)						Model No.	
			425 (800)	550 (1000)	650 (1200)	750 (1400)	875 (1600)	1100 (2000)		
18	1.0 (0.040)	0.4062	8.32	10.17	12.48	15.11	18.06	24.03	NI80-040-(†)	
20	0.81 (0.032)	0.6348	6.17	7.56	9.24	11.13	13.23	17.57	NI80-032-(†)	
22	0.64 (0.0253)	1.015	4.62	5.62	6.85	8.20	9.69	12.85	NI80-025-(†)	
24	0.51 (0.0201)	1.609	3.46	4.18	5.06	6.04	7.10	9.40	NI80-020-(†)	
26	0.40 (0.0159)	2.571	2.62	3.12	3.76	4.49	5.27	6.90	NI80-015-(†)	
28	0.32 (0.0126)	4.094	1.98	2.38	2.84	3.37	3.93	5.09	NI80-012-(†)	
30	0.25 (0.010)	6.50	1.50	1.81	2.14	2.53	2.93	3.75	NI80-010-(†)	

* Showing approximate amperes necessary to produce a given temperature, applying only to a straight wire stretched horizontally in free air.
 † Specify desired length in feet: "50" or "200". Note: This wire is not intended for use in making thermocouple elements.

Ordering Example: NI80-032-50 is a 15 m (50') spool of 20 gage bare wire.

Note: Published prices are based on market value at time of printing and are subject to change due to Nickel surcharges, Chromium and precious-metal market fluctuations.

Figure 39 – Data sheet for the test wires NI80-020-200 (Omega Engineering) and NI80-010-200 (Omega Engineering). Reproduced from Omega Engineering (2022b,c).

A.2 NCRR-17-100 (OMEGA ENGINEERING)

Resistance Heating Ribbon Wire

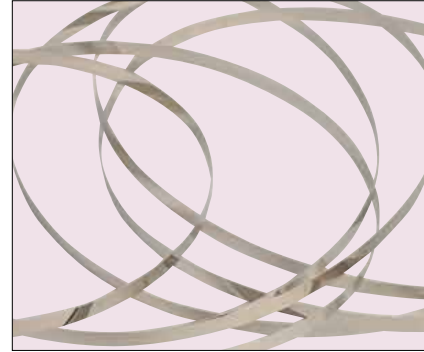
Nickel-Chromium Alloy 80% Nickel/20% Chromium



- ✓ Uniformity of Resistance
- ✓ Mechanical Stability
- ✓ Fine Surface Finish

NCRR Series
30 m (100') Spool

✓ Nickel-Chromium Wire (80Ni-20Cr) is Proven to Deliver Outstanding Performance Over Extended Periods of Time and is the Same Wire OMEGA Uses in Our Own Electrical Heating Elements

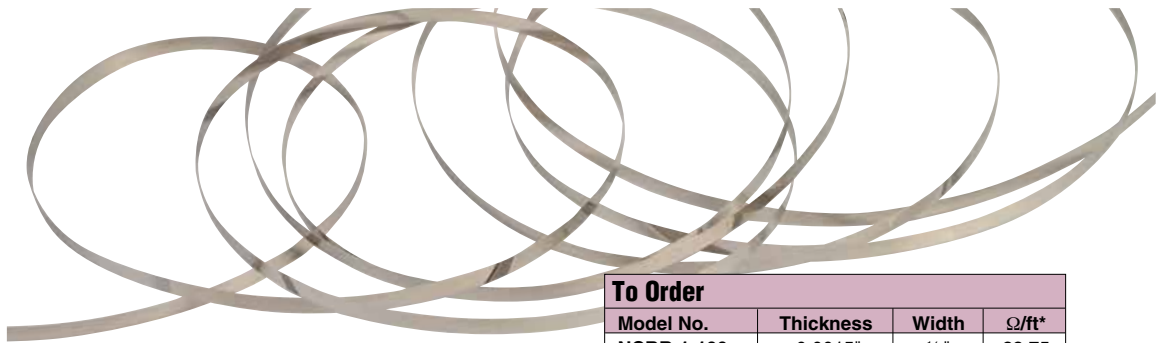


100' spool of NCRR-15-100 shown smaller than actual size.

Resistance Wire—Current vs. Temperature
Current Carrying Capacity of Straight Nickel Chromium Wire
Approximate amperes to heat a straight, oxidized wire in quiet air to given temperature

		°F	400	600	800	1000	1200	1400
		°C	205	315	427	538	649	760
AWG or B & S	Inches Diameter	Amperes						
15	0.057	7.2	10.0	12.8	16.1	20.0	24.5	
16	0.051	6.4	8.7	10.9	13.7	17.0	20.9	
17	0.045	5.5	7.5	9.5	11.7	14.5	17.6	
18	0.040	4.8	6.5	8.2	10.1	12.2	14.8	
19	0.036	4.3	5.8	7.2	8.7	10.6	12.7	
20	0.032	3.8	5.1	6.3	7.6	9.1	11.0	
21	0.0285	3.3	4.3	5.3	6.5	7.8	9.4	
22	0.0253	2.9	3.7	4.5	5.6	6.8	8.2	
23	0.0226	2.58	3.3	4.0	4.9	5.9	7.0	
24	0.0201	2.21	2.9	3.4	4.2	5.1	6.0	
25	0.0179	1.92	2.52	3.0	3.6	4.3	5.2	
26	0.0159	1.67	2.14	2.60	3.2	3.8	4.5	
27	0.0142	1.44	1.84	2.25	2.73	3.3	3.9	
28	0.0126	1.24	1.61	1.95	2.38	2.85	3.4	
29	0.0113	1.08	1.41	1.73	2.10	2.51	2.95	
30	0.0100	0.92	1.19	1.47	1.78	2.14	2.52	
31	0.0089	0.77	1.03	1.28	1.54	1.84	2.17	
32	0.0080	0.68	0.90	1.13	1.36	1.62	1.89	
33	0.0071	0.59	0.79	0.97	1.17	1.40	1.62	
34	0.0063	0.50	0.68	0.83	1.00	1.20	1.41	
35	0.0056	0.43	0.57	0.72	0.87	1.03	1.21	
36	0.0050	0.38	0.52	0.63	0.77	0.89	1.04	
37	0.0045	0.35	0.46	0.57	0.68	0.78	0.90	
38	0.0040	0.30	0.41	0.50	0.59	0.68	0.78	
39	0.0035	0.27	0.36	0.42	0.49	0.58	0.66	
40	0.0031	0.24	0.31	0.36	0.43	0.50	0.57	

Figure 40 – Data sheet for the test ribbon NCRR-17-100 (Omega Engineering) (part 1).
Reproduced from Omega Engineering (2022a).



Current Carrying Capacity of Ribbon Nickel Chromium Wire
At 648°C (1200°F) Approximate

Thickness in Inches	Width-Inches					
	1/64	1/32	1/16	3/32	1/8	3/16
	Amps					
0.0063	1.56	2.89	5.5	8.2	10.1	16.6
0.0056	1.45	2.69	5.2	7.2	9.5	15.6
0.0050	1.35	2.52	4.9	6.8	9.0	14.7
0.0045	1.26	2.38	4.6	6.4	8.5	14.0
0.0040	1.18	2.23	4.1	6.0	8.0	13.1
0.0035	1.09	2.07	3.8	5.6	7.5	12.3
0.0031	1.01	1.94	3.6	5.3	7.0	11.5
0.0020	—	—	—	—	—	—
0.0015	4	—	—	—	—	—

The current values above are based on actual tests of single strands of oxidized wire mounted in quiet air and operated at 648°C (1200°F). The tables are calculated for wire having a resistivity at 648°C (1200°F) and a total surface watts-density of 28 W per square inch.



To Order			
Model No.	Thickness	Width	Ω/ft*
NCRR-1-100	0.0015"	1/64"	22.75
NCRR-2-100	0.002"	1/64"	17.36
NCRR-3-100	0.0031"	1/64"	11.20
NCRR-4-100	0.0035"	1/64"	9.99
NCRR-5-100	0.004"	1/64"	8.68
NCRR-6-100	0.0045"	1/64"	7.12
NCRR-7-100	0.005"	1/64"	6.95
NCRR-8-100	0.0056"	1/64"	6.20
NCRR-9-100	0.0063"	1/64"	5.51
NCRR-10-100	0.002"	1/32"	9.83
NCRR-11-100	0.0031"	1/32"	5.60
NCRR-12-100	0.0035"	1/32"	4.96
NCRR-13-100	0.004"	1/32"	4.34
NCRR-14-100	0.0045"	1/32"	3.86
NCRR-15-100	0.005"	1/32"	3.47
NCRR-16-100	0.0063"	1/32"	2.76
NCRR-17-100	0.0031"	1/16"	3.17
NCRR-18-100	0.0035"	1/16"	2.81
NCRR-19-100	0.004"	1/16"	2.46
NCRR-20-100	0.0045"	1/16"	1.93
NCRR-21-100	0.005"	1/16"	1.74
NCRR-22-100	0.0056"	1/16"	1.55
NCRR-23-100	0.0063"	1/16"	1.38
NCRR-25-100	0.0031"	3/32"	2.11
NCRR-26-100	0.0035"	3/32"	1.87
NCRR-27-100	0.0045"	3/32"	1.46
NCRR-28-100	0.0063"	3/32"	0.91
NCRR-29-100	0.0031"	1/8"	1.59
NCRR-30-100	0.0035"	1/8"	1.40
NCRR-31-100	0.004"	1/8"	1.23
NCRR-32-100	0.0045"	1/8"	1.09
NCRR-33-100	0.005"	1/8"	0.98
NCRR-34-100	0.0056"	1/8"	0.88
NCRR-35-100	0.0031"	3/16"	0.88
NCRR-36-100	0.0035"	3/16"	0.78
NCRR-37-100	0.004"	3/16"	0.68
NCRR-38-100	0.0045"	3/16"	0.60
NCRR-39-100	0.005"	3/16"	0.54

* Resistance tolerance: ±5%
Ordering Example: NCRR-15-100, 100' spool of 0.0035" thickness of heating ribbon wire.

Figure 41 – Data sheet for the test ribbon NCRR-17-100 (Omega Engineering) (part 2).
Reproduced from Omega Engineering (2022a).

APPENDIX B – UNCERTAINTY ANALYSIS

According to BIPM (2008), measured variables are subject to uncertainties caused by the measurement process or the measurement equipment. Therefore, the uncertainty analysis aims to quantify the uncertainty in measurements.

BIPM (2008) classifies uncertainty sources into two categories:

- **Type A:** comprises uncertainties estimated from the statistical analysis of a series of observations;
- **Type B:** includes uncertainties obtained from other sources, such as previous measurements, manufacturer specifications, and calibration certificates;

The type A uncertainty around the measurement of a variable Y can be calculated from the measured values Y_1, \dots, Y_N as

$$\hat{U}_A = \sqrt{\frac{1}{N(N-1)} \sum_{i=1}^N (Y_i - \bar{Y})^2}, \quad (22)$$

where \bar{Y} is the average of the measured values,

$$\bar{Y} = \frac{1}{N} \sum_{i=1}^N Y_i. \quad (23)$$

In the case where the uncertainty \hat{U} is caused by different, independent sources $\hat{U}_1, \dots, \hat{U}_M$, the combined standard uncertainty can be approximated by

$$\hat{U} = \sqrt{\hat{U}_1^2 + \dots + \hat{U}_M^2}. \quad (24)$$

In particular, if the measured variable Y can be expressed as a function of other, independent variables X_1, \dots, X_M , the combined uncertainty around Y can be approximated by

$$\hat{U}(Y) = \sqrt{\left(\frac{\partial Y}{\partial X_1} \hat{U}(X_1)\right)^2 + \dots + \left(\frac{\partial Y}{\partial X_M} \hat{U}(X_M)\right)^2}, \quad (25)$$

where $\hat{U}(X_i)$ is the uncertainty around X_i .

The standard uncertainty \hat{U} represents the interval around the mean \bar{Y} that is expected to contain approximately 68.27% of the measured values and corresponds to one standard deviation of the normal distribution. However, in practical applications, increasing the confidence level of the uncertainty interval is desirable. Therefore, the standard uncertainty is multiplied by a *coverage factor* k to expand the uncertainty interval and increase the confidence level, resulting in the *expanded uncertainty*

$$U = k\hat{U}. \quad (26)$$

This work employs the coverage factor $k = 2$, which corresponds to a confidence level of approximately 95.45%.