Eur. Phys. J. C (2023) 83:1150
https://doi.org/10.1140/epjc/s10052-023-12290-4

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Theoretical Physics

A fast and time-efficient machine learning approach to dark matter searches in compressed mass scenario

Ali Celik[a]

Department of Physics, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

Received: 28 November 2022 / Accepted: 27 November 2023 / Published online: 18 December 2023

Abstract Various analyses for searching for the signature of SUSY or exotic particles have been carried out by the experiments at CERN. These analyses made use of traditional cut and count methods. While this method has yielded promising results, it has been challenging in the region where the mass difference between SUSY particles is small. Deep learning is currently widely employed in most data analysis tasks, including high energy physics, and has made significant advances in almost all fields for collecting and interpreting huge data samples. In this paper, a fast and time-efficient classification technique is proposed, utilizing machine learning algorithms to distinguish dark matter signal from SM background in compressed mass spectra scenarios at a center-of-mass energy of 14 TeV. A classification model was built in a short amount of time using 2D histograms produced with less amount of data, effectively reducing computational costs through the transfer learning of pre-trained deep models while maintaining a high level of classification accuracy.

1 Introduction

The Standard Model (SM) of particle physics is widely regarded as one of the most successful theories ever developed by mankind to date. However, it is not regarded as a complete theory for several reasons pointing to physics beyond the SM. Among these enigmas are the fact that it is unable to solve the Higgs hierarchy problem and cannot account for the presence of dark matter, whose existence is proved by the works [1–3].

The supersymmetric extension of the standard model (SUSY) [4–12] predicts a superpartner for every equivalent particle in SM that differs by a half spin. In R-parity conserving minimal supersymmetric extension of standard model (MSSM), the lightest neutralino ($\tilde{\chi}_1^0$) is the lightest supersymmetric particle (LSP), stable, weakly interacting, and thus is a candidate for dark matter. At the Large Hadron Collider (LHC), the CMS and ATLAS collaborations have been carrying out searches for supersymmetric particles. Both experiments place limits on the masses of colored supersymmetric particles. The masses of gluinos in models involving the pair production of gluinos decaying via off-shell top and bottom squarks are excluded up to $\approx$ 2.4 TeV and 2.35 TeV, respectively, for a massless LSP case [13]. The work [14] presents the combination of previously published analyses for the pair production of supersymmetric partner of top quarks in 0,1,2-leptons final state [15–17] and excludes the mass of stop ($\tilde{t}$) up to 1325 GeV for a massless neutralino; however, the largest excluded squark mass is obtained with the search [18], which excludes the mass of top squark below 1.55 TeV for a massless LSP. On the other hand, the mass limits on the electroweakly produced charginos-neutralinos are less constrained since these particles suffer from a smaller production cross section in a hadron collider. Limit on the masses of these particles are set by the work [18], reporting masses of electroweakinos, chargino($\tilde{\chi}_1^{\pm}$) and neutralino($\tilde{\chi}_2^0$) are excluded below 900 GeV.

CMS and ATLAS experiments trace for directly produced sleptons at 8, and 13 TeV in final states with di-leptons and the LSP [19–23]. As LSP leaves the detector without any trace, it contributes to missing transverse momentum, an important signal-background discriminator in SUSY searches. Assuming a mass differential of $\Delta M \leq 20$ GeV and $\Delta M \leq 60$ GeV between the slepton and the LSP, slepton signal production was investigated in events with final states having missing energy, di-lepton and an initial state radiation (ISR jet) or a pair of VBF jets at 14 TeV collision energy by the works [24–26], respectively.

The classic method to search for SUSY signal is cut-and-count method which has become a promising method thus

[a] e-mail: alicelik@mehmetakif.edu.tr (corresponding author)

far in the field of particle physics. The strategy is based on applying cuts that eliminate as much of the SM background as feasible while maintaining the maximum amount of signal events. However, it is limited by our capacity to grasp what we observe. We will need to enhance our analytical techniques to keep up with the ever-increasing volume and complexity of the data recorded by the CMS and ATLAS experiments. We can, however, overcome our limitations as humans with the help of machine learning (ML) algorithms. In fact, the ML approach is frequently a more efficient analysis method than the traditional approach, as it can process enormous datasets and return findings in a fair amount of time. ML techniques can even be better at discriminating background and signal since they can potentially find difficult-to-detect patterns in the data. Hence, machine learning approaches would enhance our ability to interpret the data, which is often multi-dimensional and complex. Work [27] investigates SUSY production in the low mass region through machine learning algorithms and compares results with the classical cut and count method used by the works [28,29]. Signal processes considered are chargino pair production, mono-Z process, slepton pair production, and chargino pair production with slepton/neutrino mediated decay. Results show that better sensitivities can be obtained with machine learning algorithms compared to the classical cut and count method.

In recent years, there has been an increasing amount of literature on the application of machine learning algorithms in SUSY searches [30–35]. In one of these [33], the authors investigated SUSY and an electrically neutral Higgs boson production in dilepton + missing energy and single lepton along with at least four jets in the final state, respectively, using machine learning algorithms. The study utilized low-level and high-level features in a combined and separate way to check the classification performance of the model as well as statistical significance value, which is used in high energy physics to assess if there is a sign for new physics.

Through the use of neural networks, work [34] investigates a number of simpler dark matter models with events containing mono-jet and missing transverse energy in the final states. However, in order to train the algorithms, the data is structured in 2D histograms rather than the classical method of passing events one by one. The histogrammed data set is fed into deep neural network (DNN) and Convolutional neural network (CNN) separately for building a model. It has been demonstrated that, when compared to DNN, CNN with 2D histograms enhances efficiency slightly. However, the primary drawbacks of these types of applications are that it necessitates more data, and training time, consequently, the usage of more hardware sources.

Characterizing Dark Matter at colliders using Machine Learning techniques has been studied by the work [35]. The focus was on the monojet and missing transverse energy (MET) channel, and a set of benchmark models beyond standard model was proposed for the study. Various representations of the data were explored, either event-by-event form or imaged versions of the kinematic distributions, which are then fed into a Logistic Regression algorithm or a Fully Connected Neural Network, Deep and Convolutional Neural Networks. All of these benchmarks were compared to each other and to the $Z + jets$ SM background. It was found that using the 2D images of the combined information of multiple events significantly improves the discrimination performance compared to the list of events with kinematical features.

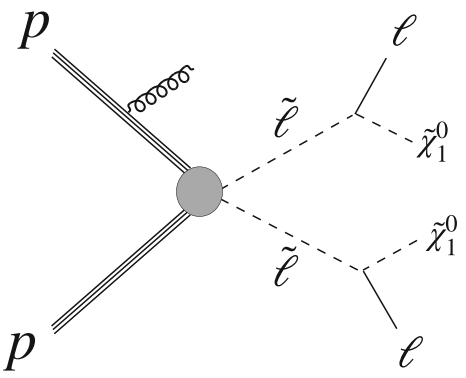

Fig. 1 Slepton pair ($\tilde{\ell}\tilde{\ell}$) production mechanism along with ISR jet emitted from one of the incident partons

The signal considered in this work is the production of a pair of left/right-handed slepton produced from Z^* or γ^* exchange in quark anti-quark/gluon-quark pair interaction together with a single extra jet emitted from one of the incident partons (see Fig. 1). Slepton can be both left/right-handed selectron or smuon. When a slepton pair is produced, both slepton decays promptly to LSP and the same flavor leptons (e^+e^- or $\mu^+\mu^-$). Final state leptons in the compressed mass spectra scenario, where the mass difference between slepton and LSP is small, are expected to be soft. Consequently, lepton reconstruction in the compressed scenario becomes a challenge, and the presence of soft products renders the signal indistinguishable from SM processes. Therefore, it is necessary to have a strongly energetic ISR jet recoiling against the sleptons, which results in an increase in the transverse momentum (p_T) of pair-produced sleptons and their decay products, consisting of a $\tilde{\chi}_1^0$ pair and same flavor opposite sign (SFOS) leptons. Namely, requiring a significant amount of missing energy coming from LSP and SFOS lepton pair and one hard jet in the final state form the signal ($pp \rightarrow \tilde{l}^+\tilde{l}^- j \rightarrow l^+l^-\tilde{\chi}_1^0\tilde{\chi}_1^0 j$).

The background process considered in this study is the production of a pair of W's (see Fig. 2 for production mechanism). WW becomes a background when each of the W's decays into a lepton and neutrino, which is nothing but a source of missing energy. Consequently, WW production mimics the SUSY signal and becomes a background. Signals and the background are generated with the lead-

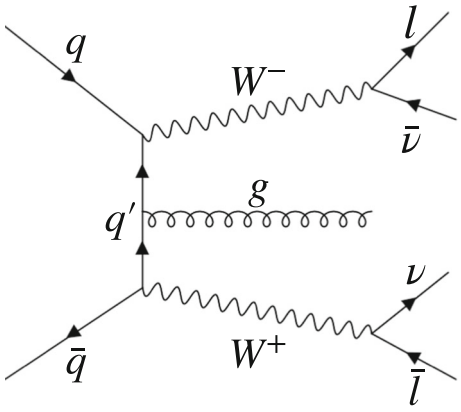

Fig. 2 Representative Feynman diagram of WW pair production followed by their decays to the same flavor leptons and neutrinos

ing order event generator MadGraph5 _aMC@NLO version 2.6.7 [36], then pushed for parton showering and hadronization into Pythia 6 [37], which is then followed by detector simulation. Detector simulation is carried out with Delphes 3 [38] and default delphes card for CMS detector [39] is used for detector response. Signal and background are generated up to two partons. MLM scheme [40] is applied to avoid double counting. SUSY spectrum generator called SUSY-HIT is utilized for generating all the parameter cards that are used as input for Madgraph in the signal production process [41].

Recently, particle physicists have shown an increased interest in the application of machine learning algorithms. However, no previous study has investigated the power of transfer learning (TL) in SUSY searches in the case of compressed spectra. Hence, in this analysis, a binary classifier was built to distinguish SM background from SUSY signal plus SM background mixture. Two machine learning algorithms, support vector machines and logistic regression, are trained with the features extracted through transfer learning. The training was done on the signal sample with small mass splitting, where the mass difference between slepton and the lightest neutralino is small ($\Delta m \equiv \Delta m(\tilde{l}, \tilde{\chi}_1^0) = m(\tilde{l}) - m\left(\tilde{\chi}_1^0\right) = 5$ GeV).

The rest of the paper has been organized in the following way. Section 2 begins by laying out the proposed method, providing details on the ML algorithms and techniques along with the signal benchmark point. The features analyzed and used for the construction of the histogrammed dataset are also explained. Section 3 is concerned with research findings and demonstrates the discrimination strength of the technique employed for the classification of signal plus SM and SM histograms. The conclusion is left for Sect. 4, which briefly summarizes and critiques the findings.

2 Methodology

The cut-and-count technique is the most typical way to extract the signal from the background. However, as shown by the distributions in Fig. 3, in some cases signal remains buried in the background, and applying cuts on the specific features may not always result in an effective signal extraction. In most recent studies, signal and background have been classified either by using conventional machine learning algorithms, deep neural networks, or convolution neural networks. However, training deep neural networks from scratch can be challenging due to several limitations, such as an imbalance in classes hindering the learning process, missing values, or unlabeled data. Moreover, training deep neural networks needs substantial computer resources, which can be costly and time-consuming. The other major problem with the method mentioned above is that it may not be very accurate and requires more data for a better result as the model is built from scratch [42–44]. However, in order to classify histograms with SM background from mixing plots of SUSY signals and SM background, machine learning algorithms are utilized after extracting features through transfer learning. This approach has several attractive characteristic features: Less training time, consequently less use of computational resources, less quantity of training data, and better at feature extraction. The strategy comprises two stages. In the first stage, some cuts characterizing the signal are applied. In the second phase, 2D histograms are generated, and then, using ML algorithms, a binary classifier is built to separate these two types of histograms.

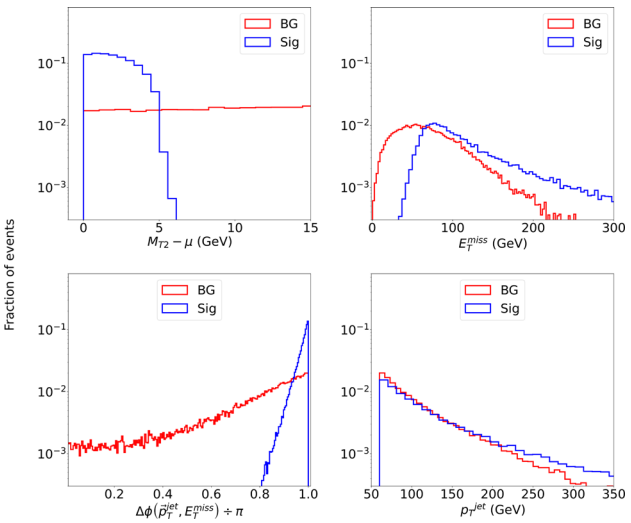

Fig. 3 Some kinematical distributions for signal and background obtained after applying the premier cuts that make the signal stands out against the background

2.1 Logistic regression

Logistic regression, despite its name, is a classification model developed by David Cox in 1958 [45]. It is commonly employed for binary and multi-class classification tasks and performs exceptionally well for linearly separable classes. This can be done by using the logistic function or known as the sigmoid function, to predict the probability of the binary outcome. Hence, the value that the logistic function gives out lies between 0 and 1. The sigmoid function used by logistic regression is shown in 1. For a given x_n, probability, $p(x_n)$, corresponds to the target y_n. Namely, when $p_n \geq 0.5$, $y_n = 1$ otherwise $y = 0$ for $p < 0.5$.

$$p = \frac{1}{1 + e^{-x}} \tag{1}$$

2.2 SVM

The Support Vector Machine (SVM) is another type of supervised learning technique that can be applied to classification and regression problems [46]. Constructing an optimal hyperplane in a multidimensional space in order to divide classes and make predictions about which classes a new example belongs to is the basic premise upon which this method is based. The best possible hyperplane is obtained by maximizing the distance between the hyperplane and the nearest data points of any class. This hyperplane is also known as a maximum-margin hyperplane.

2.3 Transfer learning

Transfer learning, also known as transfer of learning, is the process of transferring acquired knowledge and skills from one domain to another. In the context of artificial intelligence and machine learning, the goal of transfer learning is to enhance performance and reduce the amount of computational power required for solutions by levering the knowledge that has been previously learned. In recent years, transfer learning has become ubiquitous in the field of computer vision and pattern recognition, in speech recognition and recommendation engines [47–52]. Transfer learning is also widely used in the field of high-energy physics. The study [53] investigates the use of transfer learning as a new approach to train emulators for relativistic heavy ion collision simulations. The findings reveal that transfer learning is remarkably efficient and can substantially reduce the computational cost of building emulators. When training deep neural networks using simulations for a specific task like neutrino interaction classification, a significant number of simulated events is often required. Moreover, this can be computationally expensive, and the deep learning algorithm may underperform if sufficient events are unavailable. To address this issue, the study [54] examines the use of transfer learning, where a pre-trained model on generic image recognition tasks is fine-tuned using a set of simulated neutrino images for the specific task. The study used a ResNet18 model pre-trained on photographic images, fine-tuned using simulated neutrino images, and achieved an F1 score of 0.896 ± 0.002 with 100,000 training events. The paper [55] examines the potential of transfer learning techniques to develop efficient jet taggers using existing models. The primary objective was to investigate the ability of neural networks to learn the fundamental features of QCD and transfer them to a distinct task. Specifically, the study applied transfer learning to top tagging at varying transverse momentum thresholds and the tagging of boosted objects with two or three prongs, such as top quark and W boson decays. Transfer learning may be effective, particularly in the case where the data sample is insufficient to build an image classification model with high accuracy.

2.3.1 Feature extraction

Feature extraction plays a pivotal role in the domain of transfer learning, wherein the acquired information from one task or domain is utilized to enhance performance in another task or domain. In particular, pre-trained models on large datasets of images, such as ImageNet, can be used as feature extractors for other image-related tasks. This approach involves using the convolutional layers of a pre-trained depth and complex CNN models as a fixed feature extractor while removing the fully connected layers. The pre-trained model is then fine-tuned on a new domain or task by adding a new classifier on top of the extracted features. By employing this approach, the model can acquire the ability to identify and categorize objects within a novel domain using a relatively limited quantity of annotated data. This approach has been successfully applied in various domains, including medical image analysis, object detection, and natural language processing, and can significantly improve the performance of machine learning models in a wide range of applications [56–59].

2.4 Inception-v3

Inception-v3 is the third generation TensorFlow-based [60] 48-layer deep inception model introduced by Google [61]. It has been trained on over a million images from the ImageNet dataset [62] and can be utilized in computer vision tasks that require a feature extractor, such as machine learning algorithms. Inception-v3 is structured like Inception-v1, and it has 1000 image classes and can be used to extract features from an image or used as a mask to add objects to other images. The neural network has a very simple structure and is computationally efficient. This kind of feature extraction

mainly aims to detect and recognize images faster with higher accuracy.

2.5 ResNet-50

ResNet-50, a convolutional neural network architecture consisting of 50 layers, was introduced by the authors of [63] as a residual learning framework to address the issue of vanishing gradients in deep networks. Trained on the labeled subset of the ImageNet dataset, ResNet-50 has learned to recognize 1.2 million images across 1000 classes, making it a valuable pre-trained model for various computer vision tasks. The residual connections in ResNet-50 enable the flow of information from the initial layer to the final layers, facilitating the construction of deeper networks without degrading performance. As a result, ResNet-50 has achieved state-of-the-art performance in tasks such as image classification, object detection, and segmentation. While more recent models have surpassed ResNet-50's performance on the ImageNet classification task, it remains a popular and effective architecture in the field of deep learning.

2.6 Deep learning

Deep learning is an area of machine learning that makes use of neural networks in order to develop models capable of learning from data and making predictions. Neural networks are collections of linked nodes that mimic the structure and operation of the human brain. A neural network is composed of multiple layers of nodes, with each layer conducting a unique operation on the data as it traverses the network. A shallow neural network, the simple network is composed of an input layer, a hidden layer, and an output layer. When the quantity of hidden layers is expanded, the network is referred to as a deep neural network. Data is taken in by the input layer and sent on to the next processing layer. The output of one layer is used as input to the next layer, with each layer performing some computation on the output of the layer before it. Finally, the last layer contains an overall prediction or classification. The back-propagation process, which involves changing the weights of connections created between nodes, is utilized in a deep learning model in order to obtain a greater level of accuracy and reduce the amount of prediction error that occurs during training.

2.7 Convolutional neural networks

Convolutional neural networks are a type of deep learning architecture specifically engineered to effectively process and analyze various forms of data, particularly images, and videos, with the help of pattern recognition. CNNs have demonstrated exceptional performance in capturing spatial hierarchies and extracting significant features from input data, leading to their remarkable success in numerous computer vision tasks [64–67]. Designed to emulate the visual processing mechanism of the human brain, CNNs excel at recognizing and extracting intricate patterns from input data. This is achieved through the integration of multiple layers, including convolutional, pooling, and fully connected layers, which collaboratively perform the complex task of pattern recognition. Convolutional layers apply filters to localize and extract relevant features from the input data while pooling layers play a vital role in downsampling the extracted features while preserving essential information. Fully connected layers integrate the features to generate predictions or perform classifications.

2.8 Benchmark for SUSY signal

Results presented here are for the signal mass point of $m_{\tilde{l}} = 280$ GeV with $\Delta M = 5$ GeV; however, heavier slepton masses might also be scanned. In order to decouple the production of colored particles and elektroweakinos, their masses have been set to 10 TeV, which is way higher than that of interest. Right, and left-handed slepton masses are assumed to be equal, and their decay probability to the same flavor leptons are 100% $\left(\tilde{e}^+\tilde{e}^-\left(\tilde{\mu}^+\tilde{\mu}^-\right) \rightarrow e^+e^-\left(\mu^+\mu^-\right)\right.$ $\left. = 100\%\right)$.

2.9 Proposed method

Since all the attributes have varying value ranges, they do not contribute equally to the model, which is problematic for machine learning algorithms. To resolve the performance issue, the Scikit-learn's [68] "MinMaxScaler" class was employed to scale all the features to be between 0 and 1.

1. Veto on tagged hadronically decaying τ
2. Require two same flavor opposite sign leptons with $p_T >$ 10 GeV and $|\eta| < 2.4$
3. Veto events including b-jets with $p_T > 30$ GeV and $|\eta| < 2.4$
4. Require only one hard jet with $p_T > 60$ GeV and reject any events with additional jets having $p_T > 30$ GeV.

After having the events that survived the applied cuts, 2D histograms were produced from a pair combination of all the features. However, most combinations provided no information for discriminating the SM and SM+Signal case. As the m_{T2} and azimuthal angle difference between jet momentum and missing transverse energy is highly unique and discriminant for the SUSY signal, this kinematic feature pair could provide additional information to separate SM and mixing plots. Therefore, these two are utilized for the construction of 2D histograms. Each histogram in this study was constructed using a total of 25,000 events, with a bin number of

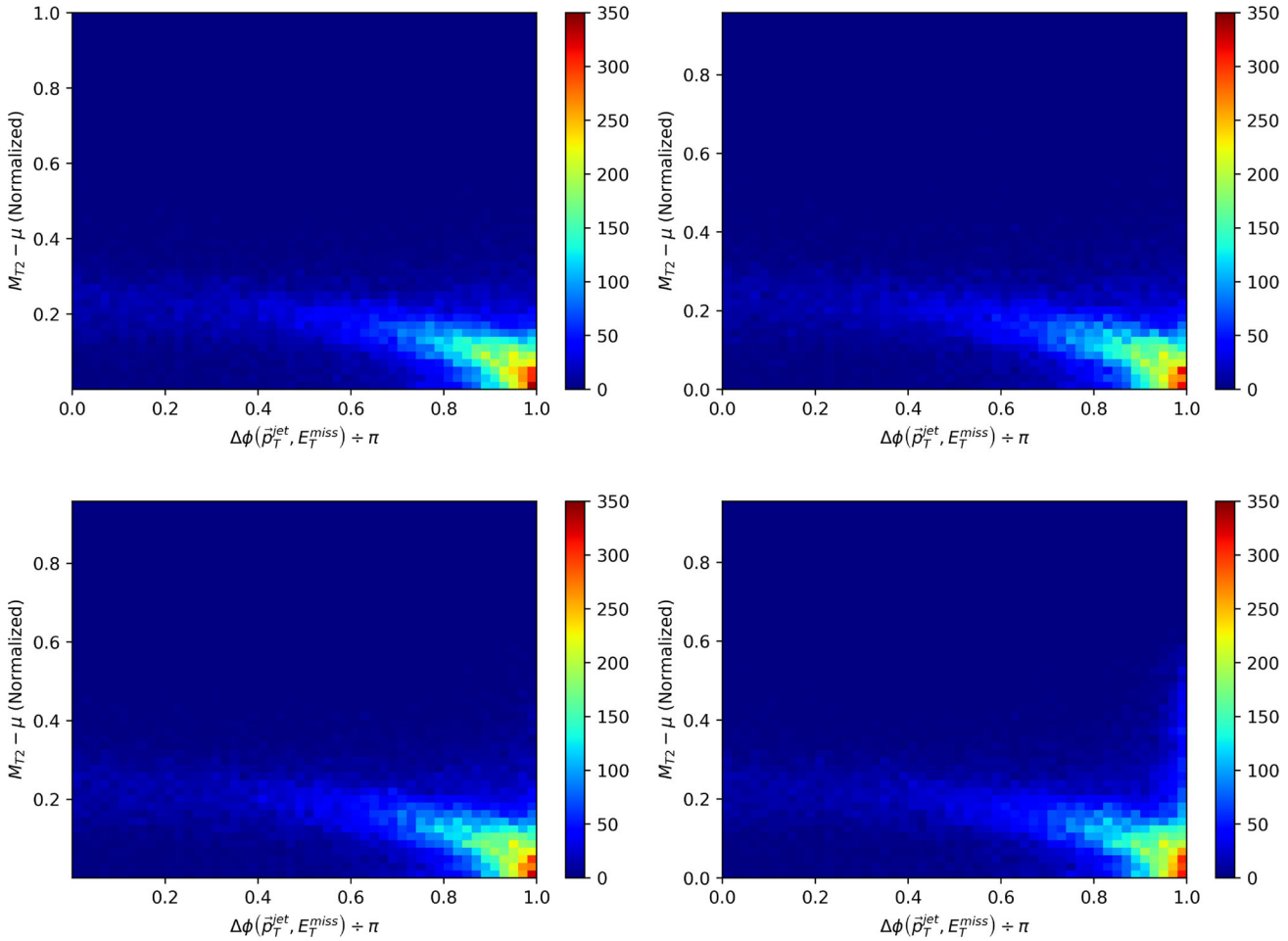

Fig. 4 2D histogram of $\Delta\phi\left(\boldsymbol{p}_T^{\text{jet}}, E_T^{\text{miss}}\right) \div \pi$ as a function of $M_{T2} - \mu$. (From top left to the right)Histogram of 25K SM events only and plot mixing SM background and signal at the ratio of $S/B = 0.001$, respectively. (From bottom left to the right) Plots from signal and SM background mixed at the ratio of $S/B = 0.01$ and $S/B = 0.1$, respectively. All the distributions maintain the total number of 25K events

50×50. While histograms including only SM background are built with 25K WW samples, other histograms mixing signal and background are constructed using the S/B ratio of 0.001, 0.002, 0.003, 0.004, 0.005, 0.007, 0.0085, 0.01, 0.02, 0.03, 0.05, 0.07, 0.1 and 0.2. For each benchmark point and each class, 1 K (10 K) histograms are produced.[1] Production of each histogram is carried out with a data augmentation technique that randomly selects the required number of samples from the total number of samples. When employing this augmentation technique, no histogram is allowed to contain the same data twice. Some example of the generated 2D histograms corresponding to $S/B = 0$, 0.001, 0.01, and 0.1 are shown in Fig. 4.

The second stage of the proposed method is to utilize the pre-trained models Inception-v3 and Resnet-50 as feature extractor modules to extract features from 2D histograms. Before feeding the images, which are in the form of a Numpy array, into the model, the pixel values were divided by 255, a vital pre-processing step. This step is crucial since it puts pixel values in the range [0, 1], enabling the model to operate effectively on a standardized input range and ensuring that each feature's scale is consistent across different samples. By leveraging the pre-trained models with these normalized image inputs, discriminative features were pulled out from the images and used for downstream tasks.

The visual feature extraction process is depicted in Fig. 5. Since the inception-v3 and ResNet-50 pre-trained models have been trained on millions of image data, they can extract features from images efficiently, resulting in improved neu-

[1] 1000 and 10,000 images are generated for transfer learning/machine learning algorithms and CNN respectively. Except for the CNN models, the number of 2D histograms produced varies depending on the S/B ratio. While 1000 images are created for most of the S/B benchmarks, the sample number is reduced accordingly with the increase of S/B. Thus, 700 created for $S/B = 0.05, 0.07$ and 500 created for $S/B = 0.1, 0.2$.

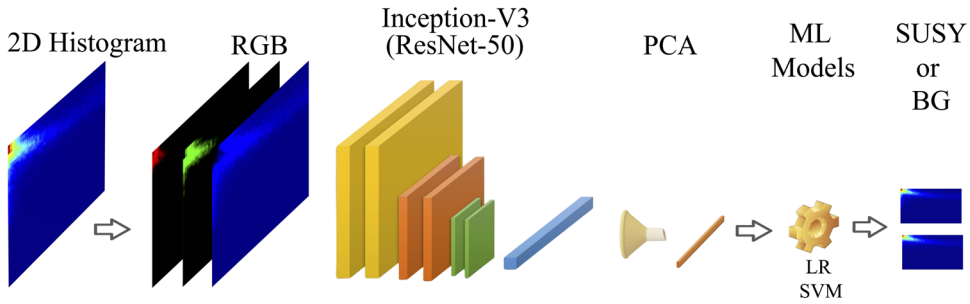

Fig. 5 The visual feature extraction process

ral network performance overall with a shorter training time. The extracted features having the size of $5 \times 5 \times 2048$ and $7 \times 7 \times 2048$ for inception and ResNet-50 respectively are then flattened and pushed into the principal component analysis (PCA) for dimensionality reduction. After employing PCA, there ended up being 2000 features for a single image, which can be denoted by $f_{PCA} \in \mathbb{R}^{1\times 2000}$, where f_{PCA} is corresponding features after employing PCA. The first three principal components explaining more than 23% of the total variance are plotted pairwise and shown in Fig. 6. PCA applied set is then used as input for machine learning algorithms SVM and LR.

In order to optimize the hyperparameters, consequently, to ensure that model performs in optimum conditions, the GridSearchCV tuning function from scikit-learn library is utilized with fivefold for transfer learning and machine learning algorithms. After fivefold grid-search, the optimum hyperparameters giving the best area under the receiver operating characteristic curve (AUC) scores for logistic regression and support vector machines are selected. Hyperparameters and their optimized values for each algorithm are tabulated

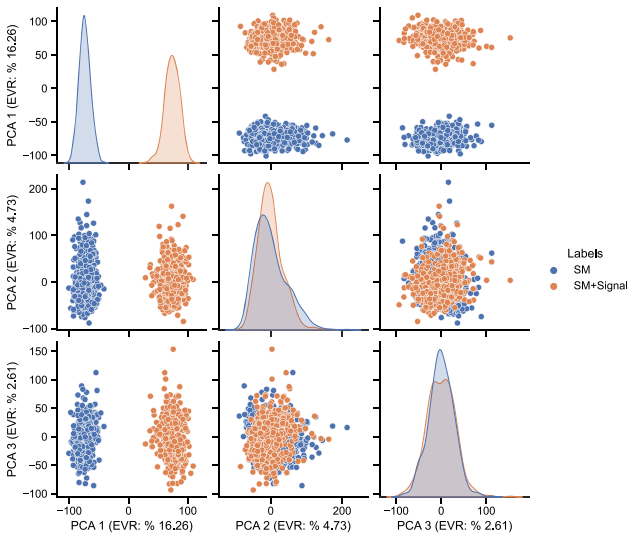

Fig. 6 Distribution of training data mapped on the first three principal components with the highest explained variance ratio (EVR). Principal components were obtained after applying PCA on the features extracted through Inception-v3 pre-trained model

Table 1 Optimum hyperparameters for SVM and LR

ML models	Hyperparameters	Inception-V3	ResNet-50	Pure SVM/LR
SVM	Kernel	*Linear*	*Linear*	*Linear*
	c	1.0	0.1	1.0
	Gamma	1.0	1.0	1.0
LR	Penalty	*l2*	*l2*	*l1*
	c	0.001	0.001	100
	Solver	*lbfgs*	*lbfgs*	*Liblinear*

and listed in Table 1. With obtained optimum parameters, two ML algorithm models for every benchmark point are built. In order to avoid overfitting the RepeatedKFold function from scikit-learn is utilized with the number of splits being five and the number of repeats being five. Namely, the whole dataset for each benchmark is split into five, four of which are used for training and one for testing. This procedure is repeated five times for each benchmark with both ML algorithms. In the binary image classification model with the convolutional neural network, various layers with different numbers of neurons were utilized to determine the optimal configuration that would yield the highest AUC score. The final model consists of several Conv2D layers with rectified linear unit (ReLU) activation, which perform feature extraction from the input images. A 3×3 filter size was used for the convolution layers to capture spatial information effectively. MaxPooling layers were employed to downsample the feature maps and reduce spatial dimensions. Dropout layers were inserted to mitigate overfitting, where a dropout rate of 0.2 was applied after each convolution layer and before the dense layers. The flattened feature maps were fed into fully connected dense layers, including a dense layer with 1024 units and ReLU activation. Finally, a dense layer with a single unit and sigmoid activation was employed for binary classification. The model has a total of 6,611,969 trainable parameters. The architecture and configuration of the model are depicted in Fig. 7.

The CNN was trained using the Adam optimizer with a learning rate set to 0.001. Binary cross-entropy was chosen as the appropriate loss function for the binary classification task (Refer to Table 2 for the complete set of hyperparameters). To evaluate the performance of the CNN model, the

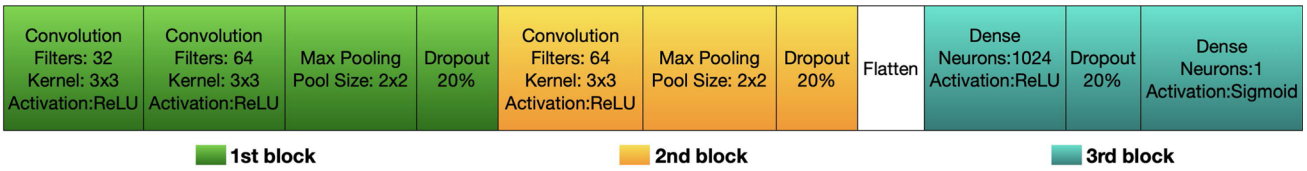

Fig. 7 The architecture of the CNN model

area under the receiver operating characteristic curve was employed as a metric, which provides a comprehensive measure of classification accuracy. The scikeras library was utilized to wrap the Keras model, enabling seamless integration with scikit-learn's cross-validation capabilities. For robust evaluation, a k-fold cross-validation approach with 10 splits was employed. The model was trained for 1000 epochs, with a batch size set to 300 during the training process.

Additionally, the performance of DNN was checked for the purpose of comparison. As the tabular data, survived the cuts mentioned in Sect. 2.9, proved to be insufficient for training the DNN models, synthetic data generation techniques were employed. To address this, approximately 25,000 more synthetic data points were generated using the Synthetic Minority Over-sampling Technique (SMOTE) [69]. Each class's tabulated data was meticulously constructed by randomly selecting background and signal events from the original tabulated dataset at varying ratios. For each ratio, 6000 "Signal+Background" and 6000 "Background Only" tabular data points were generated. Each data point corresponds to an ensemble of 1000 background events and signal event. Similarly, the "Background Only" class comprises 1000 background events in each data point. These ensemble data points were created with varying ratios of signal to background events, resulting in datasets with a shape of (6000, 1000, 12) and (6000, 1000, 2). These datasets was then subjected to PCA to reduce dimensionality while retaining a representation of approximately 93% of the original dataset's variance. The data preparation and dimensionality reduction steps ensure that the DNN model operates efficiently and effectively, with a dataset that captures the essential features while reducing computational complexity. This carefully curated dataset, along with the optimized hyperparameters, facilitated the construction of robust DNN models for varying ratios of signal-to-background events, enabling comprehensive performance evaluation.

In the binary image classification model with DNN, a systematic approach was employed for hyperparameter optimization to ensure optimal model performance. For each signal-to-background ratio, various hyperparameters were fine-tuned, including but not limited to the learning rate, the dropout rate, loss functions, etc., as listed in Table 2. This meticulous tuning was carried out using the kerastuner library [70], aiming to achieve the best model configuration. Following hyperparameter optimization, a total of five DNN models were constructed using fivefold cross-validation, each incorporating the optimized hyperparameters. To mitigate the risk of overfitting, a significant concern in deep learning, the early stopping technique was implemented, halting training if there was no improvement in validation accuracy after ten epochs. The training process was capped at 120 epochs, ensuring that the models generalize well to unseen data. Subsequently, a comprehensive evaluation was conducted on a dedicated test set using these five models. This five-fold cross-validation procedure was repeated five times for each S/B ratio, providing a robust assessment of the DNN models' performance under varying conditions.

Table 2 Training parameters for CNN and DNN models. The learning rates correspond to the S/B =0.01 case for DNN-12 (DNN-2) models, respectively

Hyperparameter	CNN	DNN
Optimizer	*Adam*	*Adam*
Activation function	*ReLu/Sigmoid*	*ReLu/Sigmoid*
Loss function	*Binary crossentropy*	*Binary crossentropy*
Batch size	300	32
Learning rate	0.001	0.0009 (0.0006)
Number of epochs	1000	120
N. of iterations	60	270
K-Fold	10	5

3 Results

The properties of the dataset utilized in this study, as well as the simulation results, are detailed in the sections that follow.

3.1 Dataset

The dataset used in this analysis for the production of 2D histograms is comprised of two classes, "1" and "0", which correspond to the SUSY signal and SM background, respectively. The total number of simulated events is 62 M, 8.5M of which belongs to the SUSY signal, while the rest is WW

Table 3 Low-level and High-level features

Low-level features	High-level features
lep1 p_T	$m_{T2} - \mu$
lep2 p_T	$\Delta\phi\left(\boldsymbol{p}_T^{l_1}, E_T^{\text{miss}}\right)$
lep1 η	$\Delta\phi\left(\boldsymbol{p}_T^{l_2}, E_T^{\text{miss}}\right)$
lep2 η	$\Delta\phi\left(\boldsymbol{p}_T^{jet}, E_T^{\text{miss}}\right)$
jet p_T	$\Delta\phi\left(\boldsymbol{p}_T^{l1}, \boldsymbol{p}_T^{l2}\right)$
jet η	
E_T^{miss}	

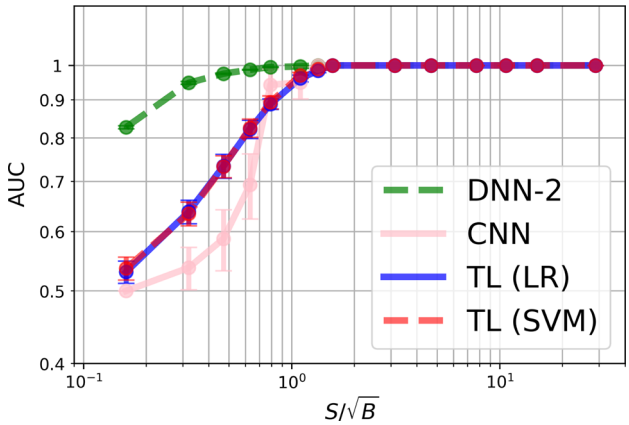

Fig. 8 AUC values for each classifier in relation to the $S/\sqrt{B}$. The presented results were derived using transfer learning (TL), as well as Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN). The green and pink lines represent DNN and CNN, while the blue and red lines denote Logistic Regression (LR) and Support Vector Machine (SVM) classifiers, respectively, when leveraging the pre-trained ResNet-50 as a feature extractor

sample. Therefore, we can express the dataset as follows:

$$D = \left\{(x_n, y_n) \mid x_n \in \mathbb{R}^{1\times M}, y_n \in \{0, 1\}\right\}$$

Here, M is 2, $n = 1 \ldots N$, and N represents the total number of raw data samples before application of any pre-selection cuts, which is 62 M. The dataset includes the kinematical variables listed in Table 3 as features. While seven of the features are low-level features, which are missing transverse momentum, transverse momentum, and pseudorapidity of leptons, and jet, the rest are high-level features, such as the azimuthal angle difference between either lepton and missing transverse, between both leptons, and between jet transverse momentum and missing transverse momentum. The most discriminating feature between signal and background is the difference between stransverse mass m_{T2} and trial mass (μ) for a given LSP mass. m_{T2} is used to calculate the lower limit on the mass of the mother particle (slepton) based on the mass and kinematics of the visible and invisible decay products. To compute the m_{T2} variable, the bisection algorithm provided by the authors of the work [71] was employed. It is presumed that final state leptons carry the same flavor and originate from the same flavor of sleptons.

Reducing the size of the datasets is essential for achieving a more manageable training time as well as preventing the consumption of excessive amounts of computing resources during the training process. Thus, before feeding the data into machine learning classifiers, a few preselection cuts listed under Sect. 2.9 were applied on the original dataset, which also helped the signal to stand out against the SM background. Some kinematical distributions obtained after applying the aforementioned cuts are displayed in Fig. 3.

3.2 Simulation findings

The simulations involving transfer learning and pure SVM/LR are conducted on a Windows 10 PC with 64-bit architecture. The PC is equipped with an i5-8250U CPU running at a clock speed of 1.8GHz and has 32GB of RAM. For feature extraction using Inception-v3 and Resnet-50, a dedicated NVIDIA GeForce 940mx GPU with a total of 6GB of RAM is utilized, while the machine learning algorithms are executed on the CPU. However, CNN and DNN models are trained on Kaggle.com since more GPU-RAM is required for building these models. For the utilization of the GPU, NVIDIA CUDA 11.2.2 toolkit is used in conjunction with the NVIDIA CUDA Deep Neural Network library v8.1.0.77. TensorFlow v2.9.1 and Keras v2.9.0 libraries are also imported.

The performance of pure support vector machines and logistic regression, as well as transfer learning, deep neural networks and convolutional neural networks, was assessed for different signal-to-background ratios. The results of these evaluations are presented in Tables 4, 5 and 6. While Area Under Curve values are used as the performance metric for this study, accuracy, recall, and F1-score for pre-trained models are also provided for reference. AUC is characterized by the area under the receiver operating characteristic curve (ROC), and its value ranges between 0 and 1. While below 0.5 suggests the model fails in classification, 1 corresponds to the model best in classification. For an AUC value of 0.7, the classifier is accepted to be successful as there is a 70% chance that the model will classify classes correctly.

Using the ROC curve that is plotted with signal efficiency (ϵ_s) as a function of background rejection ($1/\epsilon_b$), for each single benchmark point and each classifier, a mean AUC score is calculated based on fivefold cross-validation with five iterations (Fig. 8).

The AUC values obtained using DNN, pure support vector machines or logistic regression in Tables 4, 5, 6 and Fig. 8 demonstrate comparable or higher performance compared to transfer learning and convolutional neural networks. Nevertheless, the performance of both classifiers using transfer

Table 4 Performance metrics for each benchmark point evaluated with SVM, LR, CNN, and DNN. The 'DNN-12' column presents results from a model utilizing tabular dataset constructed with all twelve low and high-level features, while 'DNN-2' corresponds to results obtained from a model trained with tabular data set constructed with two features, which are also utilized for construction of 2D histogrammed data

$S/\sqrt{B}$	S/B	AUC				
		LR	SVM	CNN	DNN-12	DNN-2
0.16	0.0010	0.623 ± 0.017	0.626 ± 0.021	0.500 ± 0.000	0.656 ± 0.014	0.827 ± 0.004
0.32	0.0020	0.844 ± 0.018	0.804 ± 0.019	0.537 ± 0.035	0.898 ± 0.009	0.948 ± 0.004
0.47	0.0030	0.961 ± 0.007	0.930 ± 0.008	0.587 ± 0.055	0.962 ± 0.002	0.975 ± 0.002
0.63	0.0040	0.988 ± 0.004	0.975 ± 0.006	0.693 ± 0.069	0.986 ± 0.002	0.987 ± 0.001
0.79	0.0050	0.997 ± 0.002	0.996 ± 0.003	0.942 ± 0.047	0.994 ± 0.002	0.995 ± 0.001
1.10	0.0070	1.000 ± 0.000	1.000 ± 0.000	0.949 ± 0.047	0.998 ± 0.001	0.997 ± 0.001
1.34	0.0085	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.999 ± 0.001	1.000 ± 0.000
1.57	0.0100	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
3.13	0.0200	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
4.67	0.0300	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
7.72	0.0500	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
10.70	0.0700	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
15.08	0.1000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
28.87	0.2000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

learning exhibits minimal differences at specific benchmark points, surpassing that of CNN or falling within the range of statistical uncertainty. The observed trend in the results confirms the expectation that the models' performances enhance as the signal-to-background (S/B) ratio increases. Although both models utilizing transfer learning demonstrate strong performance across most benchmark points, neither model effectively classifies $S/B \approx 0.001$. However, an AUC value greater than 0.7 can be achieved for the $S/B > 0.003$ with transfer learning. Additionally, $S/\sqrt{B}$ was calculated for each benchmark point and presented as a function of AUC (see Fig. 8). $S/\sqrt{B}$ is used to determine if there is a new physics in the histograms for the given number of background and signal samples.

In addition to transfer learning and other machine learning approaches, in-depth training of a deep neural network was also conducted. The training process was carried out using TensorFlow and Keras on dedicated hardware, making use of a tensor processing unit (TPU) to expedite computations. The performance of the DNN was monitored through the analysis of loss trends during training. The training and validation losses were recorded at each epoch to evaluate the model's convergence and generalization capabilities. Figure 9 showcases the training and validation loss curves as a function of epoch for the model trained with all twelve features. These plots provide valuable insights into the DNN's learning behavior and its ability to adapt to the data. The DNN's performance is assessed using the AUC score. This evaluation employs a five-fold cross-validation approach, where five models are constructed and tested on the previously reserved test data. The AUC scores obtained from each of these five models on the test set are averaged and presented in Table 4.

4 Conclusion

In this work, machine learning models were built to search for new physics using 2D images constructed from a signal and a standard model background at different ratios. While the study is conducted with a single signal benchmark and a standard model background, the study might be expanded by using different signals from different models or including more signal benchmark points and/or other standard model backgrounds. The present study is designed to demonstrate that the transfer learning method on signal and background classification is efficient in terms of time and computing resources yet highly accurate.

The performance of pure SVM/LR, CNN, and DNN was also checked for the purpose of comparison. DNN and pure SVM/LR seems to yield better results for the low S/B ratio; however, training time and use of computing resources are enormous compared to transfer learning. Furthermore, in certain instances, training models solely using SVM and LR demands extensive computational resources, exceeding 27GB of RAM. The intensive resource utilization associated with this approach not only renders the application of these algorithms on larger datasets infeasible but also conflicts with the core objective of this study. Not only the computing resources used for the task were enormous, but building

Table 5 Performance metrics for every benchmark point after transfer learning with Inception-V3

$S/\sqrt{B}$	S/B	AUC		Recall		Accuracy		F-1 Score	
		LR	SVM	LR	SVM	LR	SVM	LR	SVM
0.16	0.0010	0.543 ± 0.017	0.544 ± 0.018	0.526 ± 0.029	0.528 ± 0.036	0.532 ± 0.019	0.531 ± 0.019	0.529 ± 0.019	0.529 ± 0.021
0.32	0.0020	0.567 ± 0.028	0.561 ± 0.031	0.547 ± 0.039	0.542 ± 0.048	0.552 ± 0.027	0.545 ± 0.028	0.549 ± 0.031	0.543 ± 0.036
0.47	0.0030	0.677 ± 0.024	0.670 ± 0.023	0.624 ± 0.033	0.615 ± 0.032	0.633 ± 0.030	0.621 ± 0.021	0.628 ± 0.025	0.618 ± 0.023
0.63	0.0040	0.769 ± 0.017	0.756 ± 0.016	0.681 ± 0.028	0.686 ± 0.028	0.693 ± 0.026	0.685 ± 0.019	0.689 ± 0.019	0.685 ± 0.019
0.79	0.0050	0.822 ± 0.017	0.814 ± 0.016	0.740 ± 0.025	0.733 ± 0.032	0.747 ± 0.019	0.739 ± 0.018	0.745 ± 0.020	0.737 ± 0.022
1.10	0.0070	0.908 ± 0.011	0.911 ± 0.011	0.822 ± 0.020	0.827 ± 0.025	0.829 ± 0.015	0.832 ± 0.016	0.828 ± 0.016	0.831 ± 0.018
1.34	0.0085	0.945 ± 0.008	0.950 ± 0.007	0.858 ± 0.018	0.877 ± 0.016	0.869 ± 0.011	0.880 ± 0.012	0.867 ± 0.012	0.880 ± 0.012
1.57	0.0100	0.971 ± 0.007	0.976 ± 0.006	0.908 ± 0.020	0.917 ± 0.018	0.914 ± 0.013	0.925 ± 0.011	0.913 ± 0.013	0.925 ± 0012
3.13	0.0200	0.999 ± 0.001	1.000 ± 0.000	0.984 ± 0.009	0.987 ± 0.007	0.987 ± 0.004	0.990 ± 0.004	0.987 ± 0.005	0.990 ± 0.004
4.67	0.0300	1.000 ± 0.000	1.000 ± 0.000	0.997 ± 0.003	0.999 ± 0.001	0.997 ± 0.002	0.999 ± 0.001	0.997 ± 0.002	0.999 ± 0.001
7.72	0.0500	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
10.70	0.0700	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
15.08	0.1000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
28.87	0.2000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

Table 6 Performance metrics for every benchmark point after applying transfer learning with ResNet-50

$S/\sqrt{B}$	S/B	AUC		Recall		Accuracy		F-1 Score	
		LR	SVM	LR	SVM	LR	SVM	LR	SVM
0.16	0.0010	0.530 ± 0.018	0.536 ± 0.019	0.517 ± 0.040	0.521 ± 0.027	0.518 ± 0.017	0.522 ± 0.018	0.517 ± 0.025	0.521 ± 0.022
0.32	0.0020	0.638 ± 0.023	0.634 ± 0.023	0.602 ± 0.034	0.594 ± 0.030	0.604 ± 0.021	0.598 ± 0.017	0.603 ± 0.026	0.596 ± 0.024
0.47	0.0030	0.734 ± 0.027	0.733 ± 0.025	0.660 ± 0.033	0.672 ± 0.043	0.670 ± 0.023	0.677 ± 0.025	0.666 ± 0.025	0.675 ± 0.028
0.63	0.0040	0.822 ± 0.023	0.825 ± 0.023	0.741 ± 0.023	0.747 ± 0.032	0.751 ± 0.020	0.747 ± 0.021	0.748 ± 0.018	0.747 ± 0.021
0.79	0.0050	0.888 ± 0.014	0.893 ± 0.018	0.794 ± 0.032	0.811 ± 0.030	0.806 ± 0.017	0.815 ± 0.021	0.803 ± 0.020	0.814 ± 0.024
1.10	0.0070	0.961 ± 0.010	0.970 ± 0.009	0.891 ± 0.027	0.906 ± 0.021	0.894 ± 0.015	0.915 ± 0.012	0.894 ± 0.017	0.915 ± 0.014
1.34	0.0085	0.984 ± 0.006	0.989 ± 0.004	0.931 ± 0.019	0.944 ± 0.017	0.941 ± 0.012	0.949 ± 0.010	0.940 ± 0.012	0.949 ± 0.010
1.57	0.0100	0.992 ± 0.004	0.996 ± 0.003	0.952 ± 0.018	0.971 ± 0.013	0.961 ± 0.009	0.976 ± 0.007	0.960 ± 0.009	0.976 ± 0.006
3.13	0.0200	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
4.67	0.0300	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
7.72	0.0500	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
10.70	0.0700	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
15.08	0.1000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
28.87	0.2000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

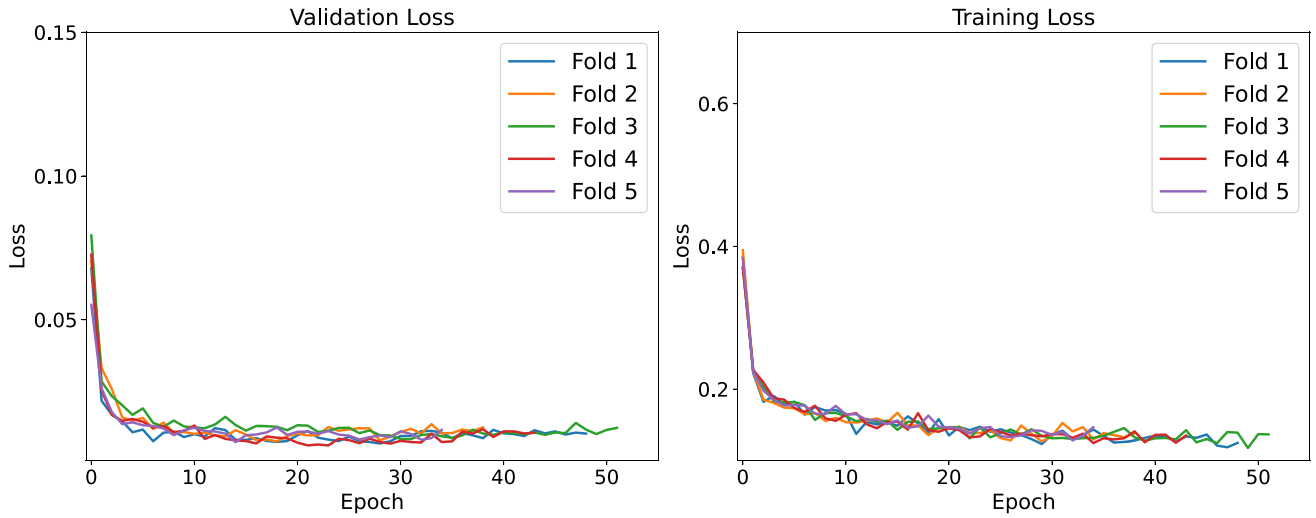

Fig. 9 Training and validation loss versus epoch for $S/B = 0.02$ with DNN-12 model. Both curves indicate the DNN's ability to adapt to the data, showcasing its capacity to capture meaningful patterns and generalize to unseen examples

models with pure SVM/LR models for some cases were also much longer compared to the proposed approach. Specifically, training time for building some models with pure LR exceeded that with transfer learning by a factor of 110. One key factor contributing to this disparity is the absence of GPU utilization for SVM and LR in the scikit-learn library. Furthermore, the developers of scikit-learn have no plans to implement GPU support for these algorithms in the near future either [72]. Thus, these algorithms may not be practical for dealing with large amounts of data as they cannot take advantage of the high computational power of GPUs.

While there exists a noticeable distinction between this study and previous works [33–35] in terms of signal model or benchmark points and backgrounds employed, direct comparison of results seems unfeasible. However, drawing from those studies, it can be inferred that convolutional neural networks or neural networks either demonstrate insignificant outcomes or necessitate a larger amount of data. In contrast, the methodology, utilizing transfer learning, offers advantages such as faster implementation and reduced computing resources, avoiding the need to train the model from scratch or gather additional data.

Starting from the feature extraction step from the images using transfer learning, followed by applying PCA and training the two models, the whole process took between 60 to 200 s and 300 to 1200s with Inception-v3 and ResNet-50, respectively. The variation in time is due to the change in sample size between benchmarks within each pre-trained model, while the time variation between the models is due to the structure of the models. Tables 5 and 6 show that ResNet-50 performs slightly better than Inception-v3 in terms of AUC, F1 score, and recall the majority of the time. This suggests that deeper networks may perform better than shallower ones. While using the Inception-v3 pre-trained model resulted in slightly worse performance than ResNet-50, it may be worthwhile to start an investigation with Inception-v3 due to its faster training time to determine if any new physics is present before committing to further investigation with other better-performance-providing pre-trained models.

Two different machine learning algorithms were employed for the classification task, with the AUC parameter chosen as the performance metric. AUC score is computed for each fold as well as for each benchmark using the ROC curve. Figure 10 illustrates the ROC curve for one of the benchmark points, $S/B = 0.01$, along with the AUC values of each fold and the mean AUC value obtained with ResNet-50. Both machine learning algorithms, SVM and Logistic Regression perform almost identically for every single benchmark point in terms of performance. Starting from the signal-background ratio equal to 0.003, the model trained with ResNet-50 begins to show promise in the classification task. With higher collision energy and increased luminosity, the expected number of signals after the aforementioned cuts will be significantly higher; as a result, the signal-to-background ratio will be higher. In such cases, the models developed, as evident from Fig. 8, will exhibit high performance. Consequently, analogous models for different SUSY models and different benchmark points can be built and used with real data in order to discover beyond standard model physics.

This study comprises an extensive examination of diverse machine learning techniques, encompassing transfer learning and deep neural networks, to address the complex task of signal-background classification in the realm of high-energy physics. The visualization of the training and validation loss trends across increasing epoch numbers, depicted in Fig. 9, offers valuable insights. These curves indicate the DNN's

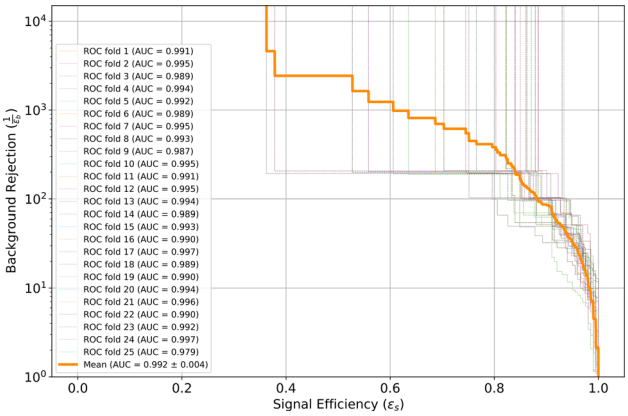

Fig. 10 ROC curves and AUC values corresponding to $S/B = 0.01$ benchmark for each fold

ability to adapt to the data, showcasing its capacity to capture meaningful patterns and generalize to unseen examples. A consistent reduction in both training and validation losses was observed throughout the training process, underscoring the effectiveness of the DNN approach for the classification of the signal+background class from only the background class.

High energy physics data often have complex structures, such as particle collision events with multiple particles and interactions. Transfer learning can be used to extract relevant discriminative features or create new high-level features from existing ones. In this context, this process might be highly important for BSM studies, particularly for detecting and separating the signal from SM backgrounds. Although pre-trained Inception-v3 and ResNet-50 are utilized as a feature extractor in this work, the study could also be expanded and performed with other pre-trained models, such as DenseNet [73] and VGG16 [74]. In addition, it may be worthwhile to try a fine-tuning approach instead of transfer learning, not for an improvement in training time but for the sake of improving accuracy.

Acknowledgements The author would like thank Tarik Kabak and Inanc Yilmaz for their technical assistance in setting up the cluster system on which portions of this work were conducted. The author also thanks Dr. Biskin and Dr. Kirbas for their valuable discussions.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: The data that support the findings of this study might be available upon reasonable request from the authors.]

Funded by SCOAP[3]. SCOAP[3] supports the goals of the International Year of Basic Sciences for Sustainable Development.

References

1. P.A.R. Ade et al., Planck2013 results. xvi. Cosmological parameters. Astron. Astrophys. **571**, A16 (2014)
2. K. Begeman, A. Broeils, R. Sanders, Extended rotation curves of spiral galaxies: dark haloes and modified dynamics. Mon. Not. R. Astron. Soc. **249**(3), 523–537 (1991)
3. R. Massey, T. Kitching, J. Richard, The dark matter of gravitational lensing. Rep. Prog. Phys. **73**(8), 086901 (2010)
4. Y.A. Gol'fand, E.P. Likhtman, Extension of the algebra of poincaré group generators and violation of p invariance, in *Supergravities in diverse dimensions*, Vol. 1 (1989)
5. D. Volkov, V. Akulov, Is the neutrino a goldstone particle? Phys. Lett. B **46**(1), 109–110 (1973)
6. J. Wess, B. Zumino, Supergauge transformations in four dimensions. Nucl. Phys. B **70**(1), 39–50 (1974)
7. J. Wess, B. Zumino, Supergauge invariant extension of quantum electrodynamics. Nucl. Phys. B **78**(1), 1–13 (1974)
8. S. Ferrara, B. Zumino, Supergauge invariant yang-mills theories. Nucl. Phys. B **79**(3), 413–421 (1974)
9. A. Salam, J. Strathdee, Super-symmetry and non-abelian gauges. Phys. Lett. B **51**(4), 353–355 (1974)
10. H.P. Nilles, Supersymmetry, supergravity and particle physics. Phys. Rep. **110**(1–2), 1–162 (1984)
11. H.E. Haber, G.L. Kane, The search for supersymmetry: probing physics beyond the standard model. Phys. Rep. **117**(2–4), 75–263 (1985)
12. S. Dawson, E. Eichten, C. Quigg, Search for supersymmetric particles in hadron–hadron collisions. Phys. Rev. D **31**(7), 1581 (1985)
13. A. Collaboration, Search for supersymmetry in final states with missing transverse momentum and three or more b-jets in 139 fb^{-1} of proton–proton collisions at $\sqrt{s} = 13$ TeV with the atlas detector (2022)
14. A. Tumasyan, W. Adam, J. Andrejkovic, T. Bergauer, S. Chatterjee, M. Dragicevic, A. Del Valle, R. Fruhwirth, M. Jeitler, N. Krammer et al., Combined searches for the production of supersymmetric top quark partners in proton–proton collisions at $\sqrt{s} = 13$ TeV. Eur. Phys. J. C Part. Fields **81**(11), 1–35 (2021)
15. A.M. Sirunyan, A. Tumasyan, W. Adam, F. Ambrogi, T. Bergauer, J. Brandstetter, M. Dragicevic, J. Erö, A. Escalante Del Valle, M. Flechl et al., Search for direct top squark pair production in events with one lepton, jets, and missing transverse momentum at 13 TeV with the cms experiment. J. High Energy Phys. **2020**(5), 1–50 (2020)
16. A.M. Sirunyan, A. Tumasyan, W. Adam, F. Ambrogi, T. Bergauer, M. Dragicevic, J. Erö, A.E.D. Valle, R. Fruehwirth, M. Jeitler et al., Search for top squark pair production using dilepton final states in pp collision data collected at $\sqrt{s} = 13$ TeV. Eur. Phys. J. C **81**, 1–30 (2021)
17. A.M. Sirunyan, A. Tumasyan, W. Adam, J.W. Andrejkovic, T. Bergauer, S. Chatterjee, M. Dragicevic, A.E. Del Valle, R. Fruehwirth, M. Jeitler et al., Search for top squark production in fully hadronic final states in proton-proton collisions at $\sqrt{s} = 13$ TeV . Phys. Rev. D **104**(5), 052001 (2021)

18. A. Collaboration, Searches for new phenomena in events with two leptons, jets, and missing transverse momentum in 139 fb^{-1} of $\sqrt{s} = 13$ TeV pp collisions with the atlas detector (2022)
19. C. Collaboration, Searches for electroweak production of charginos, neutralinos, and sleptons decaying to leptons and W, Z, and higgs bosons in pp collisions at 8 TeV. Eur. Phys. J. C **74**(9), 1–42 (2014)
20. A. Collaboration, Search for direct production of charginos, neutralinos and sleptons in final states with two leptons and missing transverse momentum in pp collisions at $\sqrt{s} = 8$ TeV with the ATLAS detector. J. High Energy Phys. **2014**(5), 1–52 (2014)
21. A. Collaboration, Search for electroweak production of supersymmetric states in scenarios with compressed mass spectra at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys. Rev. D **97**(5), 052010 (2018)
22. A. Collaboration, Search for supersymmetric partners of electrons and muons in proton–proton collisions at $\sqrt{s} = 13$ TeV. Phys. Lett. B **790**, 140–166 (2019)
23. A. Collaboration, Searches for electroweak production of supersymmetric particles with compressed mass spectra in $\sqrt{s} = 13$ TeV pp collisions with the ATLAS detector. Phys. Rev. D **101**(5), 052005 (2020)
24. Z. Han, Y. Liu, M_{T2} to the rescue: searching for sleptons in compressed spectra at the LHC. Phys. Rev. D **92**(1), 015010 (2015)
25. B. Dutta, K. Fantahun, A. Fernando, T. Ghosh, J. Kumar, P. Sandick, P. Stengel, J.W. Walker, Probing squeezed bino-slepton spectra with the large hadron collider. Phys. Rev. D **96**(7), 075037 (2017)
26. B. Dutta, T. Ghosh, A. Gurrola, W. Johns, T. Kamon, P. Sheldon, K. Sinha, K. Wang, S. Wu, Probing compressed sleptons at the LHC using vector boson fusion processes. Phys. Rev. D **91**(5), 055025 (2015)
27. M. Anderssen, Performance of deep learning in searches for new physics phenomena in events with leptons and missing transverse energy with the ATLAS detector at the LHC (2020). Accepted: 2021-01-21T23:45:39Z
28. A. Collaboration, Search for electroweak production of charginos and sleptons decaying into final states with two leptons and missing transverse momentum in $\sqrt{s} = 13$ TeV pp collisions using the ATLAS detector. Eur. Phys. J. C **80**, 123 (2020)
29. A. Collaboration, Search for an invisibly decaying Higgs boson or dark matter candidates produced in association with a Z boson in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys. Lett. B **776**, 318–337 (2018)
30. S. Caron, J.S. Kim, K. Rolbiecki, R.R. de Austri, B. Stienen, The bsm-ai project: Susy-ai-generalizing lhc limits on supersymmetry with machine learning. Eur. Phys. J. C **77**, 1–25 (2017)
31. A. Mullin, S. Nicholls, H. Pacey, M. Parker, M. White, S. Williams, Does susy have friends? A new approach for lhc event analysis. J. High Energy Phys. **2021**(2), 1–39 (2021)
32. B. Kronheim, M.P. Kuchera, H.B. Prosper, A. Karbo, Bayesian neural networks for fast susy predictions. Phys. Lett. B **813**, 136041 (2021)
33. P. Baldi, P. Sadowski, D. Whiteson, Searching for exotic particles in high-energy physics with deep learning. Nat. Commun. **5**, 4308 (2014)
34. E. Arganda, A.D. Medina, A.D. Perez, A. Szynkman, Towards a method to anticipate dark matter signals with deep learning at the LHC. SciPost Phys. **12**(2), 063 (2022)
35. C.K. Khosa, V. Sanz, M. Soughton, Using machine learning to disentangle lhc signatures of dark matter candidates. SciPost Phys. **10**(6), 151 (2021)
36. J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H.-S. Shao, T. Stelzer, P. Torrielli, M. Zaro, The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. J. High Energy Phys. **2014**(7), 1–157 (2014)
37. T. Sjöstrand, S. Mrenna, P. Skands, PYTHIA 6.4 physics and manual. J. High Energy Phys. **2006**(05), 026 (2006)
38. M. Selvaggi, DELPHES 3: a modular framework for fast-simulation of generic collider experiments. J. Phys. Conf. Ser. **523**, 012033 (2014)
39. "CMS detector DELPHES card." [Online]. https://github.com/delphes/delphes/blob/master/cards/delphes_card_CMS.tcl. Accessed 02 Sept 2022
40. M.L. Mangano et al., Matching matrix elements and shower evolution for top-pair production in hadronic collisions. J. High Energy Phys. **2007**(01), 013 (2007)
41. A. Djouadi, M. Muhlleitner, M. Spira, Decays of supersymmetric particles: the program SUSY-HIT (SUspect-SdecaY-HDECAY-InTerface) (2006). arXiv preprint arXiv:hep-ph/0609292
42. R. Mehra et al., Breast cancer histology images classification: training from scratch or transfer learning? ICT Express **4**(4), 247–254 (2018)
43. V.K. Singh et al., Segmentation and classification of multimodal medical images based on generative adversarial learning and convolutional neural networks. Ph.D. thesis, Universitat Rovira i Virgili, (2020)
44. M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, S. Mougiakakou, Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. IEEE Trans. Med. Imaging **35**(5), 1207–1216 (2016)
45. J.S. Cramer, The origins of logistic regression (2002)
46. C. Cortes, V. Vapnik, Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
47. A. Brodzicki, M. Piekarski, D. Kucharski, J. Jaworek-Korjakowska, M. Gorgon, Transfer learning methods as a new approach in computer vision tasks with small datasets. Found. Comput. Decis. Sci. **45**(3), 179–193 (2020)
48. X. Cao, D. Wipf, F. Wen, G. Duan, J. Sun, A practical transfer learning algorithm for face verification, in *Proceedings of the IEEE International Conference on Computer Vision* (2013), p. 3208–3215
49. M. Matassoni, R. Gretter, D. Falavigna, D. Giuliani, Non-native children speech recognition through transfer learning, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2018), p. 6229–6233
50. S. Ruder, M.E. Peters, S. Swayamdipta, T. Wolf, Transfer learning in natural language processing, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials* (2019), p. 15–18
51. D. Wang, T.F. Zheng, Transfer learning for speech and language processing, in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)* (IEEE, 2015), p. 1225–1237
52. O.T. Biskin, I. Kirbas, A. Celik, A fast and time-efficient glitch classification method: A deep learning-based visual feature extractor for machine learning algorithms. Astron. Comput. **42**, 100683 (2023). https://doi.org/10.1016/j.ascom.2022.100683, https://www.sciencedirect.com/science/article/pii/S221313372200097X
53. D. Liyanage, Y. Ji, D. Everett, M. Heffernan, U. Heinz, S. Mak, J.-F. Paquet, Efficient emulation of relativistic heavy ion collisions with transfer learning. Phys. Rev. C **105**(3), 034910 (2022)
54. A. Chappell, L.H. Whitehead, Application of transfer learning to neutrino interaction classification. Eur. Phys. J. C **82**(12), 1–10 (2022)
55. F.A. Dreyer, R. Grabarczyk, P.F. Monni, Leveraging universality of jet taggers through transfer learning. Eur. Phys. J. C **82**(6), 564 (2022)
56. S. Sharma, R. Mehra, Conventional machine learning and deep learning approach for multi-classification of breast cancer

histopathology images-a comparative insight. J. Digit. Imaging **33**, 632–654 (2020)
57. S. Zhou, X. Zhang, R. Zhang, Identifying cardiomegaly in chestx-ray8 using transfer learning, in *MEDINFO 2019: Health and Well-being e-Networks for All* (IOS Press, 2019), p. 482–486
58. K.S. Devan, P. Walther, J. von Einem, T. Ropinski, H.A. Kestler, C. Read, Detection of herpesvirus capsids in transmission electron microscopy images using transfer learning. Histochem. Cell Biol. **151**, 101–114 (2019)
59. E. Deniz, A. Şengür, Z. Kadiroğlu, Y. Guo, V. Bajaj, Ü. Budak, Transfer learning based histopathologic image classification for breast cancer detection. Health Inf. Sci. Syst. **6**, 1–7 (2018)
60. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software available from tensorflow.org
61. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), p. 2818–2826
62. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2009), p. 248–255
63. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), p. 770–778
64. S. Lawrence, C. Giles, A.C. Tsoi, A. Back, Face recognition: a convolutional neural-network approach. IEEE Trans. Neural Netw. **8**(1), 98–113 (1997)
65. D. Ciregan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), p. 3642–3649
66. Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, M. Chen, Medical image classification with convolutional neural network, in: *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)* (2014), p. 844–848
67. N. Sharma, V. Jain, A. Mishra, An analysis of convolutional neural networks for image classification. Procedia Comput. Sci. **132**, 377–384 (2018)
68. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
69. N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
70. T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, et al., "Kerastuner." (2019). https://github.com/keras-team/keras-tuner
71. C.G. Lester, B. Nachman, Bisection-based asymmetric M_{T2} computation: a higher precision calculator than existing symmetric methods. J. High Energy Phys. **2015**(3), 1–16 (2015)
72. "Will you add GPU support?" [Online]. https://scikit-learn.org/stable/faq.html#will-you-add-gpu-support. Accessed 12 Apr 2023
73. G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), p. 4700–4708
74. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint arXiv:1409.1556