IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 21, 2021, accepted May 3, 2021, date of publication May 7, 2021, date of current version May 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078184

iSPLInception: An Inception-ResNet Deep Learning Architecture for Human Activity Recognition

MUTEGEKI RONALD[1], ALWIN POULOSE[2], AND DONG SEOG HAN[2], (Senior Member, IEEE)
[1]School of Computer Science and Engineering, Kyungpook National University, Daegu 41566, South Korea
[2]School of Electronic and Electrical Engineering, Kyungpook National University, Daegu 41566, South Korea

Corresponding author: Dong Seog Han (dshan@knu.ac.kr)

This work was supported by the Technology Innovation Program (Development of vehicle ICT convergence advanced driving assistance system and service for safe driving of longterm driving drivers) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea) under Grant 20003519.

ABSTRACT Advances in deep learning (DL) model design have pushed the boundaries of the areas in which it can be applied. The fields with an immense availability of complex big data have been big beneficiaries of these advances. One such field is human activity recognition (HAR). HAR is a popular area of research in a connected world because internet-of-things (IoT) devices and smartphones are becoming more prevalent. A major research goal of recent research work has been to improve predictive accuracy for devices with limited computational resources. In this paper, we propose iSPLInception, a DL model motivated by the Inception-ResNet architecture from Google, that not only achieves high predictive accuracy but also uses fewer device resources. We evaluate the proposed model's performance on four public HAR datasets from the University of California, Irvine (UCI) machine learning repository. The proposed model's performance is compared to that of existing DL architectures that have been proposed in the recent past to solve the HAR problem. The proposed model outperforms these approaches on several metrics of accuracy, cross-entropy loss, and F_1 score on all the four datasets. The performance of the proposed iSPLInception model is validated on the UCI HAR using smartphones dataset, Opportunity activity recognition dataset, Daphnet freezing of gait dataset, and PAMAP2 physical activity monitoring dataset. The experiments and result analysis indicate that the proposed iSPLInception model achieves remarkable performance for HAR applications.

INDEX TERMS Human activity recognition, deep learning, inception module, Inception-ResNet, time-series classification.

I. INTRODUCTION

Human activity recognition (HAR) as an area of research has been advancing for decades due to its societal benefits when applied in real-life human-centric applications. Potential application areas span elderly-care, healthcare, smart homes, athletics, and abnormal activity monitoring [1]–[5]. As the world's aging population continues to grow, the demand for applications that continuously monitor the physical well-being of a user has grown in tandem. The rapid progress in semiconductor technology and the low cost of sensors like gyroscopes, accelerometers, and magnetometers that are lightweight, small, and consume very little power have made them easy to incorporate in most modern electronic devices like smartwatches and smartphones. This has further motivated HAR-related research to become more skewed towards becoming sensor-based [2]. This, however, has not limited relevant research into other methods of obtaining data for activity recognition like vision-based [6] and Wi-Fi-based approaches [7], [8] but these have been associated with several limitations.

Vision-based approaches utilize a vision sensor, a camera, for example, that is placed in the environment where the subject performs their daily activities. This renders them constrained by factors like illumination, location, obstruction, occlusion, privacy concerns among others. On the other hand, Wi-Fi-based approaches leverage advances in Wi-Fi signals and improved public wireless infrastructure to detect the

The associate editor coordinating the review of this manuscript and approving it for publication was Giuseppe Desolda.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

change in patterns of Wi-Fi signals reflected by a human body to recognize what activity a user is performing. Their limitations include the computational complexity and resource drain on these systems, signal fluctuations, interference of Wi-Fi signals, and obstacles in the environment.

The biggest challenge for sensor-based HAR systems is feature extraction. The ability to infer a specific class to which a sensor stream belongs has been a subject of immense research. Whereas in the past, statistical machine learning (SML) approaches (both spectral and frequency-based) have been proposed, recent works have built on what Krizhevsky *et al.* [9] proposed in trying to address an image recognition problem. Deep learning (DL) in general and convolutional neural networks have evolved a great deal especially in the context of HAR.

In the intelligent signal processing lab (iSPL), our key research goal in activity recognition is to achieve the best predictive accuracy and least cross-entropy loss when differentiating between activities performed by a user. This is in addition to performing HAR on devices with very low computational resources, by users that are not domain experts with very small datasets to work with. Whereas the majority of successful HAR research has so far focused on basic human activities, up-and-coming research explores more complex activities. To be able to move towards complex human activities, however, it is desirable to exhaustively perform well in classifying basic activities by evaluating existing publicly available datasets and creating a benchmark on which HAR approaches should be compared.

In this paper, we propose a DL model, iSPLInception, that builds upon the work from [10] to perform the human activity recognition task. Szegedy *et al.* [10] continued refining their work on the use of Inception modules and also introduced a key component that has given very large models the ability to converge faster and perform extremely well, residual connections that were proposed by He *et al.* [11]. In doing so, we build a very deep and wide network that is not only computationally efficient, converges very fast but is also scalable.

The main contributions made by this paper are summarized below;

- We propose iSPLInception, a DL model for classifying human activities. The proposed DL model is based on the Inception-ResNet model and we have built it to work on a human activity recognition task achieving significant results when compared to existing approaches.
- We establish a benchmark for future research of four publicly available datasets. These were retrieved from the University of California, Irvine (UCI) machine learning repository. The datasets used are; the UCI HAR using smartphones, Opportunity activity recognition, Daphnet freezing of gait, and PAMAP2 physical activity monitoring datasets.
- We compare our approach's performance to existing research on the four datasets. We compare it to a vanilla long short-term memory (vLSTM), stacked long short-term memory (sLSTM), convolutional neural network (CNN), CNN-LSTM, and bidirectional LSTM (BiLSTM) networks.

The remainder of this paper is organized as follows: Section II explores the previous studies undertaken that relate to our research and applicative context. Our proposed model architecture is discussed in Section III. Section IV presents the experimental setup and the results with specific observations we obtained during this study. Finally, a discussion and conclusion around our proposed approach, the results, and future works are presented in Section V.

II. RELATED WORK

The multitude of application areas within which HAR is applicable has made it a well-researched field. In the intelligent signal processing lab (iSPL) of Kyungpook national university (KNU), our work focuses on wearable/handheld sensor-based recognition of human activities. HAR based on signal generating sensors is geared towards classifying the body's physical motions and gestures from which one can infer a user's action or behavior. HAR has been used in healthcare [2], [12], surveillance-based security systems [5], sports activity monitoring [13], mobile and edge computing [14], and ambient assisted living (AAL) [15]. The widespread use of smartphones and watches that are equipped with a multitude of sensors has led to the need for customized modern methods to monitor physical activities, habits, and behavior.

A. HUMAN ACTIVITY RECOGNITION

Chen *et al.* [6] put together an impressive collection of work on HAR. HAR aims at differentiating among common activities of daily life (ADL) like running, sitting, laying, riding, exercising, etc. The approaches to solving the HAR problem have evolved from traditional machine learning (ML) algorithms that were dominant in the early 2000s [3] to deep learning (DL)-based approaches [1], [2], [7], [16], [17]. Every solution in several technological fields that involves big data currently relies on or is transitioning towards using DL. Several problems in speech recognition, natural language processing, signal processing, and image recognition have found optimal solutions from deep learning and neural networks. More recently, several papers [1], [2], [7], [17] have been published featuring DL models that can extract and select features, recognize human activities, and even make use of the recognized activities in real-world applications.

This paper focuses on HAR that relies on motion sensors and therefore, we limit ourselves to reviewing work on motion sensor-based approaches. Most modern smartphones are equipped with motion sensors and thus making this type of HAR the most widely researched. Chen *et al.* [6] consider activity recognition to be a complex process that is characterized by four basic tasks which include (1) choosing and deploying appropriate sensors to objects and environments to monitor and capture a user's behavior along

with the state change of the environment, (2) collecting, storing and processing perceived information through data analysis techniques and/or knowledge representation formalisms at appropriate levels of abstraction; 3) creating computational activity models in a way that allows software systems/agents to conduct reasoning and manipulation, and 4) to select or develop reasoning algorithms to infer activities from sensor data. They furthermore suggest that each task has a raft of methods, technologies, and tools that are available for use. It is often the case that the selection of a method used for one task is dependent on the method of another task.

From the above tasks, researchers have classified HAR into; vision-based [6], sensor-based [2], [13], [16], and radio frequency-based [7], [8]: considering the sensing mechanisms and tools used, data-driven vs knowledge-driven activity recognition: in which models or mechanisms that interpret sensor data to infer activities can either rely on the patterns learned from data (bottom-up) or from rich prior knowledge in the domain of interest (top-down) [6]. Our work is sensor-based and data-driven whose aim is to establish a verifiable benchmark for HAR datasets.

The ideal process for HAR uses data from common sensors embedded in smartphones, watches, or standalone sensors; accelerometer, gyroscope, magnetometer, and linear accelerometer [1], [18]. The data is preprocessed, windowed, and then loaded into a machine learning model for analysis, a process we refer to as extract transform load (ETL), ready for the deep learning model to infer from a list of already learned human activities. The statistical machine learning (SML) techniques that were used in the past to distinguish among different human activities [3], [18], [19] made HAR out to be difficult and complicated, requiring domain knowledge and a lot of varied data to recognize patterns.

Damaševičius *et al.* [20] use the Jaccard distance to tell different activities apart, Lin *et al.* [3] create a social computer game named Fish'n'Steps to link a player's daily footstep count to the growth and activity of an animated virtual fish in a fish tank using a pedometer's step count. Jain and Kanhangad [19] use the histogram of gradient (HOG) and centroid signature-based Fourier descriptors to generate features from sensor signals that are then classified using multiclass support vector machine (SVM) and *k*-nearest neighbors (*k*-NN) classifiers. Anguita *et al.* [18] using smartphone signal data, engineer features both in time and frequency domains and use a multiclass SVM to group them into the 6 classes of walking, walking-upstairs, walking-downstairs, sitting, standing, and laying down. We use a variation of the **UCI HAR** smartphone dataset [18], in our work. This work has been extended by Garcia-Gonzalez *et al.* [21] with a new dataset that only contains four activities of active, inactive, walking, and driving and data from the accelerometer, gyroscope, magnetometer, and global positioning system (GPS) of the smartphone. Thu and Han [22] address the presence of postural transitions within another variation of this dataset and propose a way to deal with them when trying to improve the classification of the activities.

Chavarriaga *et al.* [23] while presenting the outcomes from the Opportunity challenge used the nearest centroid classifier, *k*-NN, quadratic discriminant analysis, and linear discriminant analysis. The dataset from this work we refer to as the **Opportunity** dataset and was first proposed by Roggen *et al.* [24]. Reiss and Stricker [25] in establishing a benchmark dataset for physical activity monitoring focused on 12 protocol activities. They used the Weka toolkit with 5 different SML classifiers to obtain different benchmarks that include the *intensity estimation task*; where classes are obtained based on the metabolic equivalent (light, moderate and vigorous), the *basic activity recognition task* which has 5 activity classes (lie, sit/stand, walk, run and cycle), the *background activity recognition task* that has the basic activities and the *other* class and finally the *all activity recognition task* that defines a separate class for each of the 12 activities. In our work, we refer to this dataset as the **PAMAP2** dataset. Bachlin *et al.* [26] introduces a wearable system whose aim is to assist walking of Parkinson's disease patients along with the **Daphnet** dataset. This dataset contains *freeze* or *no freeze* classes with data from wearable acceleration sensors placed on both legs and the hip.

B. DEEP LEARNING

The SML techniques that have been used in the past required deep domain understanding to implement, let alone design. The biggest challenge of relying on hand-crafted features is the failure to find an accurate characteristic or group of characteristics that differentiate all the activities. In recent years, however, more advanced DL-based classification pipelines have been built for HAR [2], [16], [17], [41]. DL significantly differs from SML in that it makes it more convenient to extract and classify complex data even with data coming from heterogeneous sensor sources and modalities [16]. With DL approaches, the idea is such that raw or preprocessed data is fed to a trained DL model at one end and the classification result comes out at the other end including internal feature generation [37]–[40].

Some of the challenges for the systems based on this approach face include; the need for a lot of training data for them to be extensively usable, intra-class variations: where activity has different characteristics when performed by different users, inter-class similarities: where activities share a lot of characteristics, like jogging and running might be perceived as similar from a sensor's perspective, and data imbalances where data in one or more classes is significantly more than that of other classes. Some solutions that have been proposed include transfer learning (TL) [1] a phenomenon that allows models trained on a source dataset, which is ideally bigger and more generalized, is retrained on a smaller or more specific target dataset to address this. The ability of a DL model to perform extremely well when the data for training and testing is taken from the distribution (or user) has made this very viable. Additionally, using hybrid

networks optimized for different sensor modalities comes in handy when working with HAR problems [35], [36].

C. INCEPTION-ResNet

Since Krizhevsky *et al.* [9] and their "AlexNet" network from the ImageNet LSVRC-2010 contest, the convolutional neural network (CNN) has become the go-to architecture for not only image recognition tasks but also any task that uses complex data that would lead to complex relations being used to distinguish one class from another. Szegedy *et al.* [27], through GoogLeNet (Inception-v1), goes even further to improve on the ease with which relations can be discovered while achieving a smaller number of parameters yet being a lot more accurate [27]. The key idea introduced by the Inception-v1 architecture is the inception module that has undergone several improvements since 2015 when it was released.

The GoogleNet network is comprised of these modules stacked upon each other, with max-pooling layers that halve the resolution of the grid. Adding 1×1 convolutional layers to the network increases the depth and this is used heavily in Inception-ResNet [10]. However, in [10], 1×1 convolutions are used both as dimensionality reduction modules to remove computational bottlenecks, which would have otherwise limited the size of the built networks and therefore not just increasing the depth, but also the width of the resultant networks without significantly penalizing the network on performance.

Inception-v2 and v3 were both introduced in [28]. Ioffe *et al.* [29] introduced batch normalization which was used in Inception-v2 and Szegedy *et al.* [28] added factorization in v3. The fourth iteration, Inception-v4, was a further improvement to v3 [10]. In [10], they also introduce Inception-ResNet which we have modified and made use of in this research work. Although Szegedy *et al.* [10] does not seem to support the view that residual connections introduced by He *et al.* [11] are inherently necessary for training very deep convolutional models, more than one research has come out to make this claim. The residual connection enables the proposed model to grow significantly while maintaining its superior performance and not overfit like other deep neural networks.

In our approach, we use this Inception-ResNet network to perform the HAR task and we have only made use of a few parts of the original architecture from [10] as elaborated in Section III. Our work significantly differs from all the works mentioned here, although shares slight similarities with the works by Xu *et al.* [16] and a bit more structural similarities with [30]. They proposed a model that combines a modified Inception model with a gated recurrent unit (GRU), which is a variant of the CNN-LSTM architecture. On the other hand, Ismail Fawaz *et al.* [30] built an "**InceptionTime**" model to address general time series classification (TSC) problems from the UCR time series archive and uses the 85 datasets therein to compare and redefine what the HIVE-COTE algorithm has been able to achieve on the same datasets.

III. PROPOSED iSPLInception MODEL

Although previous deep learning approaches have advanced the performance of HAR systems, they have been unable to improve performance as well when they scale and thus led to overfitting. We propose the intelligent signal processing lab Inception (iSPLInception) that builds on the successes realized by the Inception-ResNet model in other fields of deep learning to address the activity recognition task. The iSPLInception model proposed here is premised on the Inception-ResNet model, especially the residual connection and Inception module. Fig. 1 shows our implementation of the Inception module using TensorFlow's Keras API.

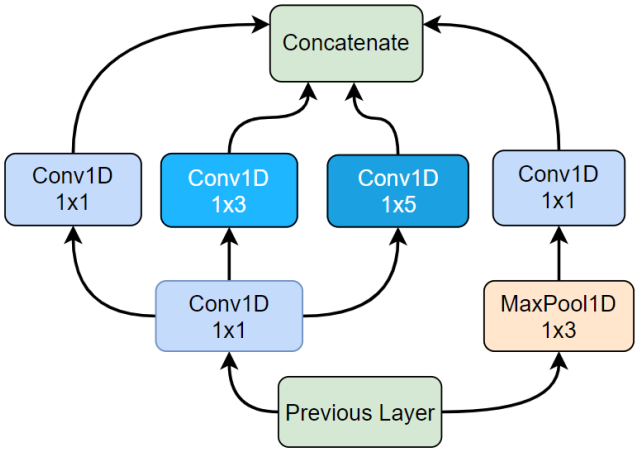

FIGURE 1. Modified Inception module used in the proposed iSPLInception model.

Inception modules are quite useful when building models that are very deep and wide. In each Inception module, we run convolutions with different kernel sizes in parallel after which we concatenate the output from these parallel operations. The immediate layer before this block provides the input. This is connected to both a 1×1 1-dimension convolution layer (Conv1D) and the MaxPool1D, a 1×3 max pooling operation. This 1×1 Conv1D is a cheap operation that works as a dimensionality reduction layer for the input features and is much cheaper to work with by removing the extra channel as presented in [10] and [28]. We refer to this 1×1 Conv1D as a bottleneck that reduces the input channels to 1. It acts as input for 3 more convolution layers; a 1×1, a 1×3, and a 1×5 Conv1D. These are either connected to the concatenation layer or the residual node depending on the level of the network on which a particular layer is. The kernel sizes of 1×3 and 1×5 Conv1D layers are determined by the maximum kernel size hyperparameter. The MaxPool1D layer is connected to a 1×1 Conv1D layer that is also connected to either the residual node or concatenation layer.

Fig. 2 illustrates the full view of our proposed iSPLInception model. Being a HAR problem, the input is a group of signals with a uniform length (window size). The input array is of the shape (batch size, window size, number of signals). The number of samples per input sequence, S, can

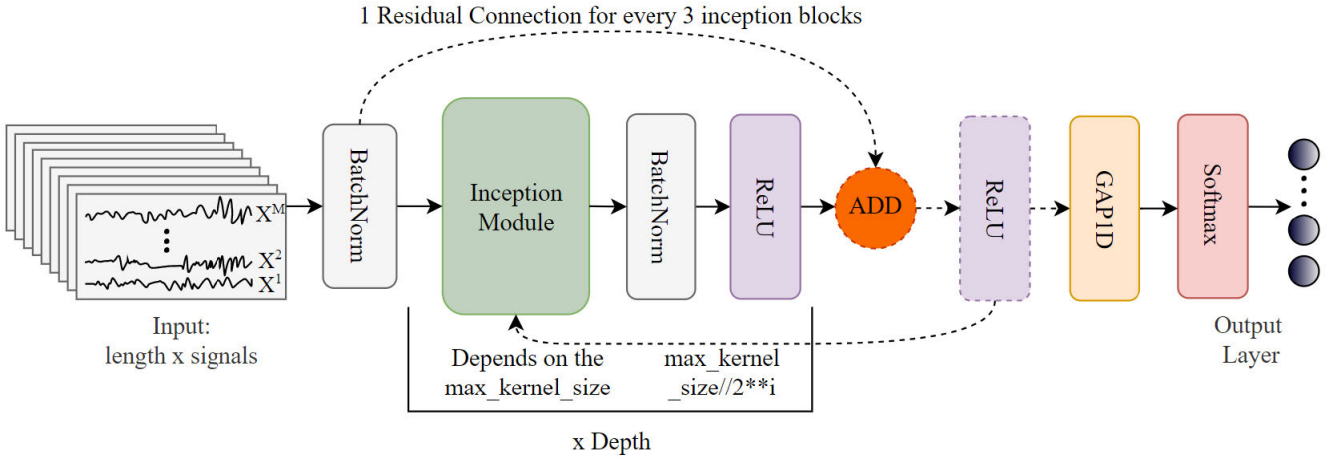

FIGURE 2. The architecture of our proposed iSPLInception network.

be described as

$$S = win\_size \times sampling\_rate \tag{1}$$

where *win_size* is the size of each window in seconds, and *sampling_rate* is in Hz. For example, the UCI HAR smartphone dataset has signals from 6 sensors sampled at 50 Hz (20 milliseconds (ms) per sample). We create 2.56 s overlapping windows which translate to 128 data samples per window.

From (1), the input shape to the model depends entirely on the dataset we are dealing with. The window size also varies from one dataset to the other which allows the iSPLInception model to be utilized to solve a variety of HAR TSC problems. The input signals are passed on to the batch normalization (BatchNorm) layer which acts as our "*previous layer*" for the first Inception block from Fig. 1. The output for the Inception module is then passed on to another BatchNorm layer before a rectified linear unit (ReLU). The number of Inception module, BatchNorm, and ReLU layers used is determined by the depth hyperparameter. Our proposed model was been built to be scalable and so we can have as many inception blocks as needed making this model wider and deeper without significantly increasing model size in terms of parameters. For every 3 of these blocks, we add a residual connection that connects the input layer of the first inception module, in Fig. 2's case a BatchNorm layer, to the output of the activation function which is combined using the Add layer.

The output from this residual operation is activated using a ReLU activation layer. The presence or absence of the residual connection entirely depends on the depth hyperparameter of the proposed model. The second ReLU activation depends on whether the Inception block has a residual connection or not. The residual connection is referred to as a shortcut and allows us to feed fresh information through a skip connection [10] to succeeding layers. In [10], we learn that succeeding layers can learn as much information from the same input data as preceding layers or even more. The output from the final ReLU activation function goes through 1-dimensional global average pooling (GAP1D) as presented by Ismail Fawaz *et al.* [30].

A. CLASSIFICATION LAYER

We use a sigmoid classification layer with binary cross-entropy for the Daphnet dataset and softmax classification with categorical cross-entropy for the rest of the datasets. The general cross-entropy loss, CE_{general}, can be described as

$$CE_{\text{general}} = -\sum_{i}^{C} t_i \log s_i \tag{2}$$

where t_i and s_i are the true values and the model scores (predicted values) respectively for the i^{th} class in classes C. Usually, an activation function like Softmax or Sigmoid is applied to the scores s_i before the CE Loss computation and thus we use $f(s_i)$ to refer to the activations [31].

While working with the Daphnet dataset, a binary classification problem, we use the sigmoid activation function which squashes an input vector in the range (0, 1) and is applied independently to each element of the scores s, s_i. The sigmoid activation function, $Sigmoid(x)$, can be denoted by

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

where $Sigmoid(x)$ is the sigmoid function for the input x and e is the Euler's number. By combining (2) and (3), we obtain the binary cross-entropy (sigmoid) loss, CE_{sig}, which can be defined by

$$\begin{aligned} CE_{\text{sig}} &= -\sum_{i=1}^{C'=2} t_i \log\{f(s_i)\} \\ &= -t_1 \log\{f(s_1)\} - (1 - t_1) \log\{1 - f(s_1)\} \end{aligned} \tag{4}$$

where it is assumed that there are two classes: C_1 and C_2. $t_1[0, 1]$ and s_1 are the groundtruth and the score for C_1, and $t_2 = 1 - t_1$ and $s_2 = 1 - s_1$ are the groundtruth and the score for C_2. More information on the sigmoid activation function and loss can be found in [31].

On the other hand, the softmax function, σ, also squashes a vector in the range (0, 1) and all the resulting elements add up to 1. The softmax function cannot be computed independently for each input value because it depends on all the elements of this vector. The input values can be positive, negative, zero, or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. A model has trained to output a probability over the C classes for each input vector. The softmax activation for the i^{th} input vector, $\sigma(\vec{s})_i$, is denoted by

$$\sigma(\vec{s})_i = \frac{e^{s_i}}{\sum_{j=1}^{C} e^{s_j}} \tag{5}$$

where $\vec{s}$ is the input vector, e^{s_i} is the standard exponential function for the input vector, C is the number of classes in the multi-class classifier, e^{s_j} is the standard exponential function for the output vector. The term at the bottom is the normalization term. It ensures that all the output values of the function will sum to 1 and each be in the range (0, 1).

We combine this softmax activation with CE_{general} from (2) to get the categorical cross-entropy (softmax) loss. This softmax loss, CE_{softmax} can be denoted by

$$CE_{\text{softmax}} = -\log\left(\frac{e^{s_p}}{\sum_{j}^{C} e^{s_j}}\right) \tag{6}$$

where e^{s_p} is the softmax score for the positive class p. In multi-class classification where the labels are one-hot encoded, only the positive class C_p keeps its term in the loss. There is only one element of the target vector t in (2) which is nonzero, $t_i = t_p$. This is the reason why (6) is softmax loss [31].

In addition to using softmax loss for the UCI HAR, PAMAP2, and Opportunity datasets and sigmoid loss for the Daphnet dataset, other key distinctions on the way we treated these datasets is expanded upon in the experiment setup section of this paper.

B. MODEL IMPLEMENTATION

The proposed model architecture was implemented in TensorFlow [32] using the Keras API. TensorFlow is an interface for expressing machine learning algorithms and an implementation for executing such algorithms. We use TensorFlow 2.4.0 which supports eager execution and accelerates training on GPUs.

Keras is a high-level API for TensorFlow that enables users to easily build artificial neural networks by abstracting all the complexities. More details on the deep learning machine and technology used are summarized in Table 1.

TABLE 1. Deep learning machine specifications.

Metric	Description
Processor	Intel Core i7-6900K , 8 Cores
Clock Frequency	3.20 GHz per core
Memory	80.0 GB
Graphics Cards	2 TITAN Xp - 30 Cores each 1 TITAN X (Pascal) - 28 Cores
GPU Memory	12 GiB each
Compute Capability	6.1
Core Clock Range	1.531 - 1.582 GHz
Operating System	Windows 10 Pro - x64
Python	3.7.0
TensorFlow & Keras	2.4.0

IV. EXPERIMENTS AND RESULTS

In this section, we first introduce the experimental setup in terms of how we configured the different models with their respective parameters, the different configurations of the datasets, and further analysis of the machine specifications on which we ran the experiments. Next, we present and discuss the results we obtained for each dataset and compare the results for the different models.

A. EXPERIMENTAL SETUP

To thoroughly evaluate the performance of the proposed model on the HAR problem and against that of the other models, we used four publicly available and benchmark datasets which we describe in more detail as we present their results. Fig. 3 illustrates the architectures we implemented for the CNN [33], CNN-LSTM [2], vanilla LSTM [2], stacked LSTM [2], and BiLSTM [22] networks. These models have been proposed to perform TSC especially the HAR task in the past with resounding success and are elucidated upon in a subsequent subsection.

All the models were trained on the same train, validation, and test sets for consistency and more meaningful comparison. The publicly available datasets we used have been online for a long time and several research works have made use of them with each claiming superiority over the other [18], [19], [23], [25], [26].

In our experiments, we made specific configurations to these datasets that would result in a more uniform benchmark for deep learning applications and newer approaches to compare with. For example, since all datasets have subject-specific information, we have ensured that data from the same subjects has not been used in both training and testing sets. However, when working with the Daphnet dataset, we used some data from the same user in both the testing and validation sets although it was from different experiments/drills. This was due to the rarity of data that belongs to the freeze class. Some additional considerations have been mentioned in subsequent subsections.

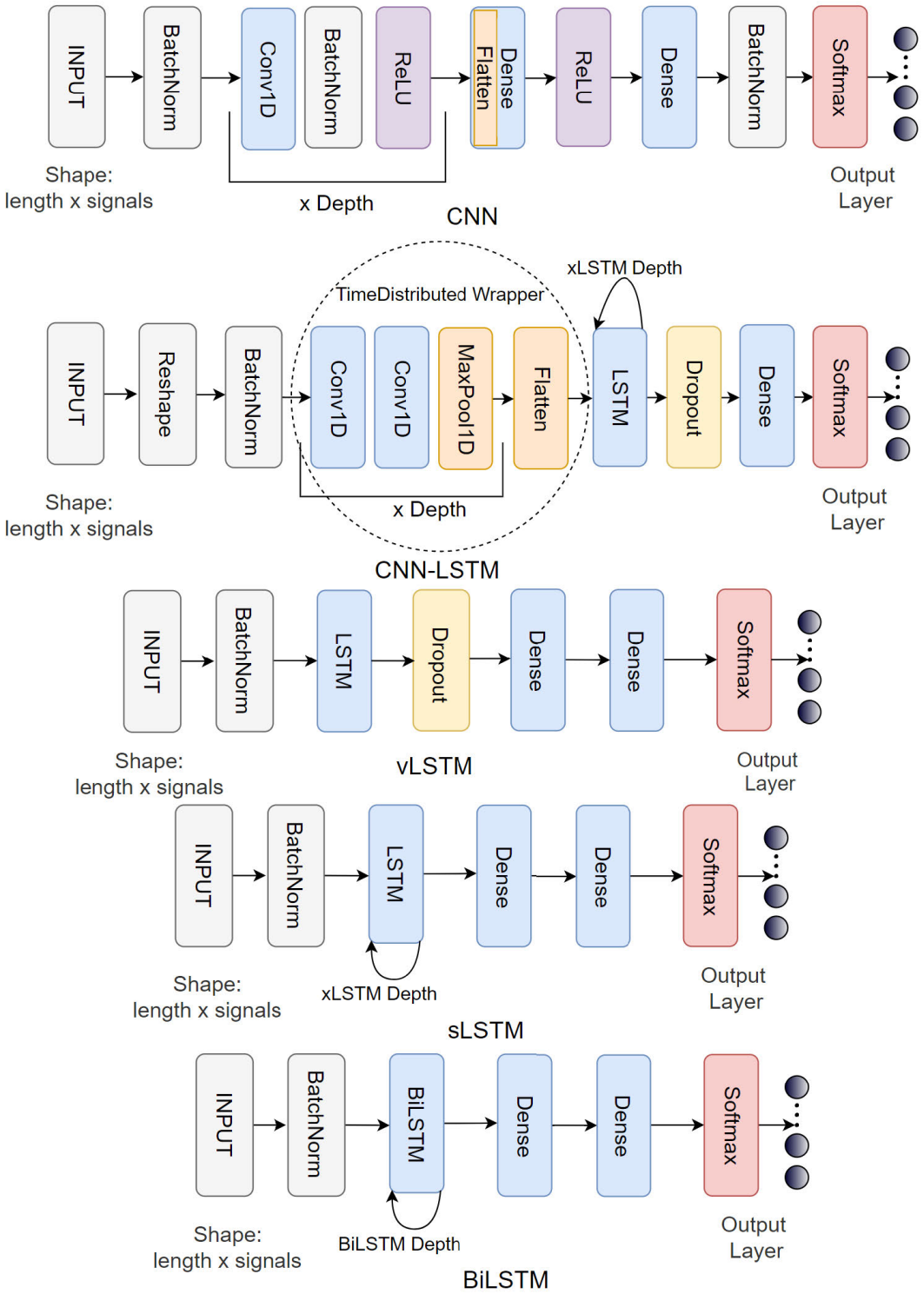

FIGURE 3. Architectures of the CNN, CNN-LSTM, vanilla LSTM, and stacked LSTM models used in our experiments.

B. MODEL CONFIGURATIONS

In Fig. 3, at the very top is the CNN model we built using the Keras sequential API. Just like our iSPLInception model, it has a BatchNorm layer just after the Input layer. This is followed by an ensemble of Convolution blocks that have 1 Conv1D layer followed by a BatchNorm layer and a ReLU activation layer. The number of these blocks is defined by the depth parameter of our network. In our experiments, we chose 5 as the depth of our CNN. The output from the last block is flattened using the Flatten layer and then passed on to a

Dense layer with 100 neurons, a ReLU activation, another Dense layer with neurons equal to the number of classes in the dataset, a BatchNorm layer, and finally an Activation layer that is either Softmax or Sigmoid.

The architecture we adopted for the CNN-LSTM network is quite like those from [1] and [2]. We build it using the sequential API and is comprised of an Input layer that is identical for all the models used. The input is however reshaped to a specified number of steps (*n_steps*), length of each step (*length*), and the number of signals (*signals*) for each window. For example, a typical input to the model is comprised of the length of the window and the number of signals (channels). We split each window of x time steps into sub-sequences. x should be divisible by *n_steps*. For the UCI HAR dataset, we set the $n\_steps = 4$ since $x = 128$ and 128 is divisible by 4. The use of a reshape layer enables us to accommodate any dataset with uniform inputs as well as the test set without having to preprocess the input to the network.

The reshaped input is then fed to a BatchNorm layer, followed by the Convolution blocks. Each convolution block is comprised of 2 Conv1D, one after the other, and a MaxPool1D layer. After the final convolution block, we flatten the output and feed the flattened output to the LSTM block. We include a dropout of 0.3 before a 100 neuron Dense layer after which is a Dense classification layer with the number of neurons equal to the number of classes and an activation function that is either Sigmoid for binary classification or SoftMax for multi-class classification.

One thing to note is the use of the time distributed wrapper for the convolution blocks. According to [32] and the TensorFlow documentation, this wrapper allows us to apply a layer to every temporal slice of an input. The input should be at least 3D, and the dimension of index one will be the temporal dimension. If we consider a batch of 64 data samples from the UCI HAR dataset, where each sample has a window length of 128 data points and 9 signals (channels), we will convert this to 4-time slices/steps each, of 32 data points. The batch input shape is $(64, 4, 32, 9)$. Since this kind of layer applies the same instance of Conv1D to each of the timestamps, the same set of weights are used at each timestamp.

The architecture we adopted for the vLSTM, a simple vanilla LSTM network, is also shown in Fig. 3 and features a single LSTM layer with 128 hidden neurons that are followed by a Dropout layer, then a 100 neuron Dense layer. This culminates into a classification layer which is a Dense layer whose number of neurons is equivalent to the number of classes of the dataset and is the same for all models.

The stacked LSTM network, sLSTM, just like the name suggests, is one where we just stacked together a bunch of LSTM layers whose depth is determined by the *depth* parameter which is then followed by the fully connected 100 neurons Dense layer and a classification layer that is just like that of the other architectures described previously.

Finally, the BiLSTM network has a BatchNorm after the Input layer that is followed by the single Bidirectional LSTM network that uses a concatenation merge mode. This is followed by a 100-neuron dense layer. The output layer is the same as those described before for other models.

The source code and more details of how the models were configured and set up can be found on the accompanying GitHub repository in [34].

C. PERFORMANCE MEASURES

Naturally, the overall human activity datasets that are collected in a natural environment are often imbalanced among classes. Whereas some of the classes may contain many samples, other classes might have only a few samples. Among our 4 datasets, the UCI HAR dataset is a balanced dataset with the class with the lowest number of samples in the train set having 13% while the largest class having 19%. The same goes for the validation and test classes. However, the activities of the Opportunity dataset are extremely imbalanced, where the *Drink from Cup* class accounts for more than 23% of the training data while the *Close Drawer 2* only has 2% of the training data. PAMAP2 has some imbalances but Daphnet is the most imbalanced of the 4 datasets where the *No Freeze* class accounts for over 91% of the data with the *Freeze* class having just 9%.

When evaluating the performance of a model on a dataset, accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observations to the total observations. One may be let to think that, if we have high accuracy then the proposed model is the best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positives and false negatives are almost the same. Therefore, you must look at other parameters to evaluate the performance of your model. The accuracy, A, of the model is computed as

$$A = \frac{TP + TN}{TP + FP + FN + TN} \tag{7}$$

where TP and FP are the number of true and false positives, respectively, and TN and FN correspond to the number of true and false negatives.

When any classifier predicts the classification accuracy of each class, the classes with higher amounts of data achieve very high accuracy which is not the case for smaller classes. The overall classification accuracy is not an appropriate index for performance evaluation [16]. F1 score, F_1, considers the correct classification of each class as equally important. It considers both the precision and the recall of each class to compute the score. Precision, P, is the ratio of correctly predicted positive observations to the total predicted positive observations and can be defined as

$$P = \frac{TP}{TP + FP} \tag{8}$$

Recall or sensitivity, R, is the ratio of correctly predicted positive observations to all observations in the actual class

and can be defined as

$$R = \frac{TP}{TP + FN} \tag{9}$$

The F_1 score is the weighted average of P and R which counters class imbalances by weighting classes according to their sample proportion and can be defined as

$$F_1 = 2 * \frac{R * P}{R + P} \tag{10}$$

where R is the recall and P is the precision.

In our experiments, we evaluate the models based on the model's accuracy, categorical or binary cross-entropy loss, the F_1 score from (10), and the number of total parameters generated by the model. Several models generate lower loss values due to their small sizes which put bigger, deeper, and more complex models at a disadvantage.

D. UCI HAR DATASET

The UCI HAR smartphone dataset was proposed by Anguita *et al.* [18] and is built from the recordings of 30 subjects performing physical/basic activities of daily life (BADL) with a waist-mounted smartphone that has embedded inertial sensors. The goal is to classify 6 activities with data recorded from triaxial linear acceleration and triaxial angular velocity at a constant sampling frequency of 50 Hz. The dataset contains three static postures (*standing, sitting, lying*), and three dynamic activities (*walking, walking downstairs and walking upstairs*). The data was sampled in fixed-width sliding windows of 2.56 seconds with a 50% overlap which resulted in 128 readings per window.

The signals were preprocessed for noise reduction with a median filter and a 3rd order low-pass Butterworth filter with a 20 Hz cutoff frequency. The acceleration signal, which has gravitational and body motion components, was separated using another Butterworth low-pass filter into body acceleration and gravity. In total, we obtained 9 signals/channels to act as input to the DL models.

To provide a benchmark worth reproducing and for comparing with future work, using the *datareader.py* file from [34], we split the dataset into the train, test, and validation sets based on the subjects as shown in Table 2. This was meant to ensure consistent and verifiable results for all our models and the generalizability of the trained models across all users. The minimum and maximum class percentages indicate a well-balanced dataset across the 6 classes.

Table 3 shows the results we obtained from the UCI HAR dataset using the different models.

Our iSPLInception network attains the highest test accuracy of 95.09% and F_1 score of 95% when compared to the other models. It is closely followed by the CNN-LSTM network that attains a 0.61% lower accuracy and 94% F_1 score. The categorical cross-entropy loss from (6) is 0.1761 which is better than that of the CNN-LSTM model that attained 0.2137. For the model size, however, the proposed model has over 1,300,000 parameters which are significantly high when compared to the vanilla LSTM network with just about

TABLE 2. Splitting the UCI HAR dataset and data disparity.

Set	Subjects	Samples	Min Class	Max Class
Train	1, 3, 5, 6, 7, 8, 11, 14, 15, 16, 17, 19, 21, 22, 23, 25, 26, 27, 28, 29, 30	7352	13.4%	19.1%
Test	2, 9, 10, 13, 18, 24	1956	14.5%	18.2%
Validation	4, 12, 20	991	13.7%	19.2%

TABLE 3. Performance of the different models on the UCI HAR dataset.

Model	Accuracy	Loss	F_1 Score	Parameters
CNN [33]	91.67	0.2977	92	424,158
CNN-LSTM [2]	94.48	0.2137	94	3,501,830
vLSTM [2]	90.80	0.3296	91	**84,198**
sLSTM [2]	91.82	0.3995	92	610,534
BiLSTM [22]	93.91	0.2777	94	167, 654
iSPLInception	**95.09**	**0.1761**	**95**	1,327,754

85,000 parameters although has far fewer parameters than the CNN-LSTM network that has over twice the number of parameters.

We illustrate the training and validation accuracy and loss for the iSPLInception model in Fig. 4. We trained the proposed model and the other models for a maximum of 350 epochs with early stopping patience set to 100 epochs and used a learning rate scheduler to iteratively reduce the learning rate when the training plateaus. Fig. 5 shows how the models performed on the UCI HAR dataset with our iSPLInception model being better than the other models.

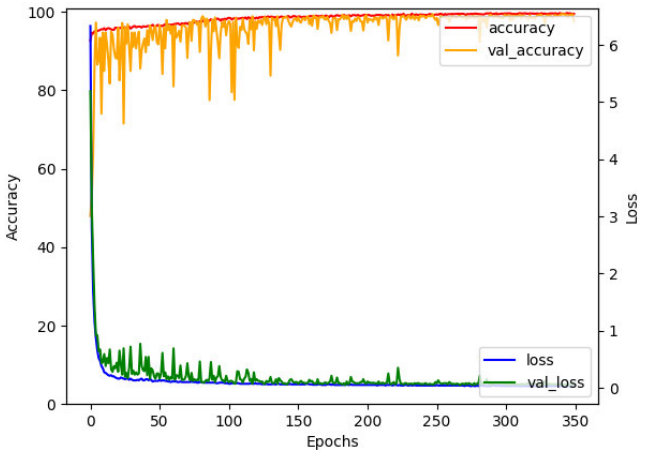

FIGURE 4. Performance of the iSPLInception model on the UCI HAR dataset.

In Fig. 6, the confusion matrices from our 6 models are compared and show that the proposed model performs

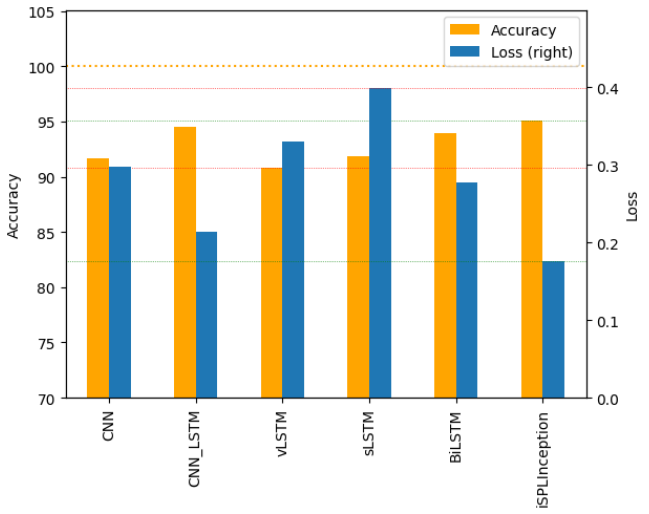

FIGURE 5. Comparing the various models on the UCI HAR dataset.

better in telling the different classes apart. In most of the models, the sitting and standing classes are easily mixed up because of their shared characteristics. Our iSPLInception model however manages to tell them apart with more precision.

E. OPPORTUNITY DATASET

Roggen *et al.* [24] proposed the Opportunity activity recognition dataset which comprises naturalistic activities which were collected in a sensor-rich environment using 72 environmental and body sensors. It comprises recordings of 12 subjects using 15 networked sensor systems, with 72 sensors of 10 modalities, integrated with the environment, in objects, and on the body. These characteristics make it well suited to benchmark various activity recognition approaches.

We considered data from only the inertial measurement units that belong to the columns between 38 and 134. We took data from the triaxial accelerometer, gyroscope, magnetometer, and others but excluded the quaternion measurements. This resulted in 77 signals (channels) as our input. The data was sampled at 30 Hz and we extracted 3-second windows that resulted in 90 samples per window from (1).

The Opportunity dataset is initially an 18-class multi-class classification problem, but we exclude the null class, an additional label, and thus use 17 classes. These are listed in Table 4. Of these, "*Close Drawer 1*" and "*Close Drawer 2*" are the smallest with just about 2.5% each of the entire dataset whereas the "*Drink from Cup*" class has the most data.

This dataset is considered imbalanced due to the disparity in the distribution of the data for the classes. Table 5 summarizes the subject data, the number of samples in the train test and validation sets and the percentage of the smallest and largest classes in the dataset.

In Table 6, we summarize the results from training our models on the Opportunity dataset.

TABLE 4. Classes we used for the opportunity dataset.

Open Door 1	Close Drawer 1
Open Door 2	Open Drawer 2
Close Door 1	Close Drawer 2
Close Door 2	Open Drawer 3
Open Fridge	Close Drawer 3
Close Fridge	Clean Table
Open Dishwasher	Drink from Cup
Close Dishwasher	Toggle Switch
Open Drawer 1	

TABLE 5. Splitting the opportunity dataset and data disparity.

Set	Subjects	No_Samples	Min	Max
Train	S1-2, S1-3, S1-4, S1-5, S1-Drill, S2-1, S2-3, S2-4, S2-5, S3-2, S3-4, S3-5, S4-1, S4-2, S4-3, S4-Drill	3,045	2.6%	23.1%
Test	S2-2, S2-Drill S3-1, S4-5	1,189	2.3%	22.1%
Validation	S1-1, S3-3, S3-Drill, S4-4	1,087	3.1%	19.9%

TABLE 6. Performance of the different models on the opportunity dataset.

Model	Accuracy	Loss	F_1 Score	Parameters
CNN [33]	80.24	0.8037	80	310,513
CNN-LSTM [2]	81.41	0.5734	81	3,444,205
vLSTM [2]	76.79	0.7988	77	**120,397**
sLSTM [2]	80.82	0.6405	81	271,557
BiLSTM [22]	79.90	0.6559	80	238,669
iSPLInception	**88.14**	**0.4790**	**88**	1,354,789

Our iSPLInception model performs significantly better on this dataset when compared to the other models. With an F_1 score of 88% which is 7% higher than that of the CNN-LSTM and stacked LSTM models that each attained an 81% score. We achieve an 88.14% model accuracy and a 0.4790 model loss which are much higher than that of the stacked LSTM and CNN-LSTM networks. The proposed model parameters are only lower than those of the CNN-LSTM model like with the UCI HAR dataset.

We illustrate the training and validation accuracy and loss for the iSPLInception model in Fig. 7. We trained the proposed model and the other models for a maximum of 350 epochs with early stopping patience set to 100 epochs and used a learning rate scheduler to iteratively reduce the learning rate when the training plateaus. Fig. 8 shows how

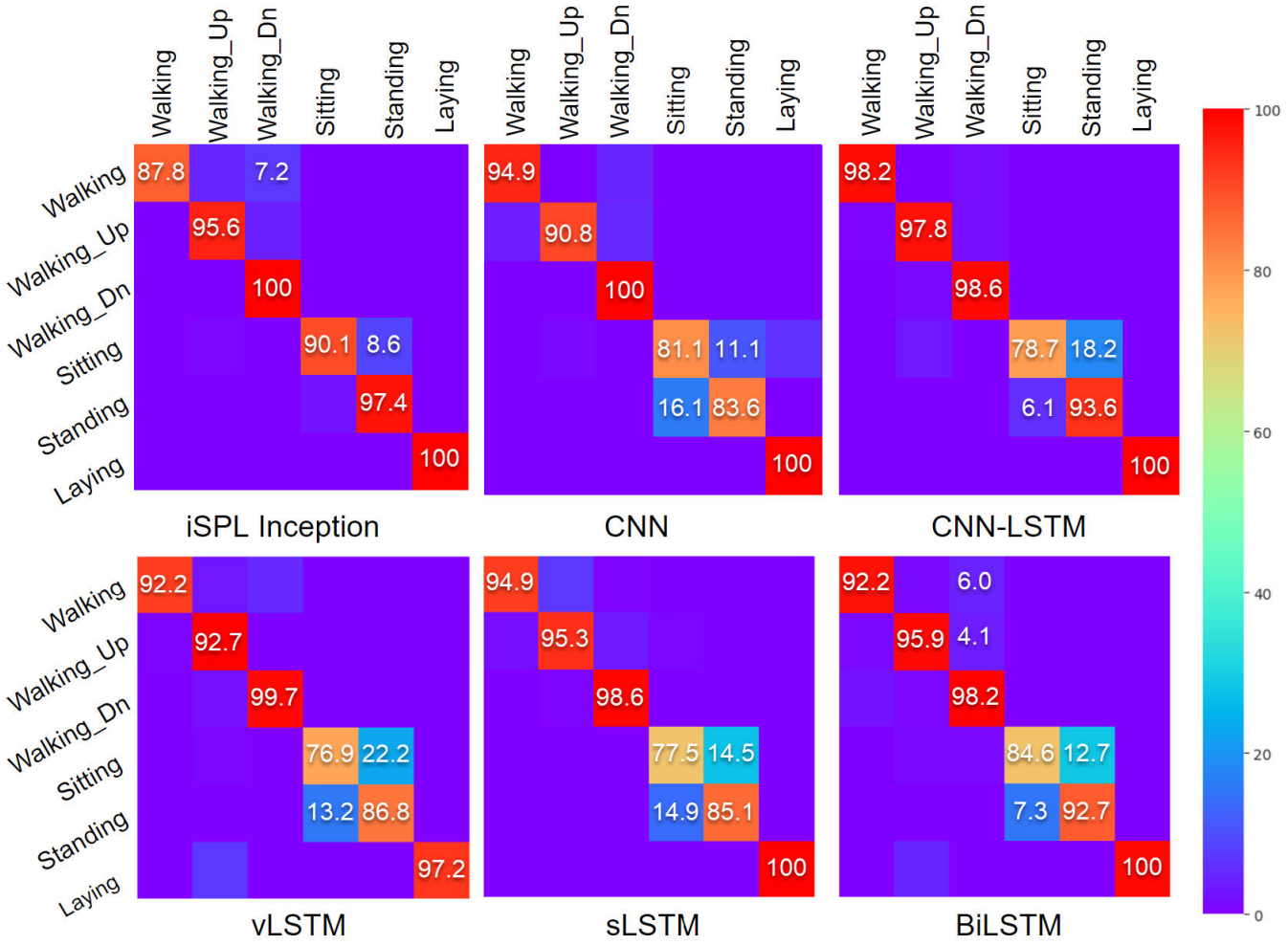

FIGURE 6. Confusion matrices for all the models on the UCI HAR dataset.

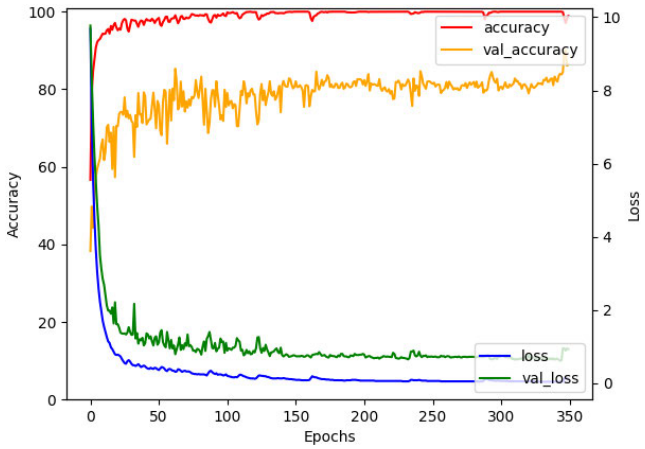

FIGURE 7. Performance of the iSPLInception model on the opportunity dataset.

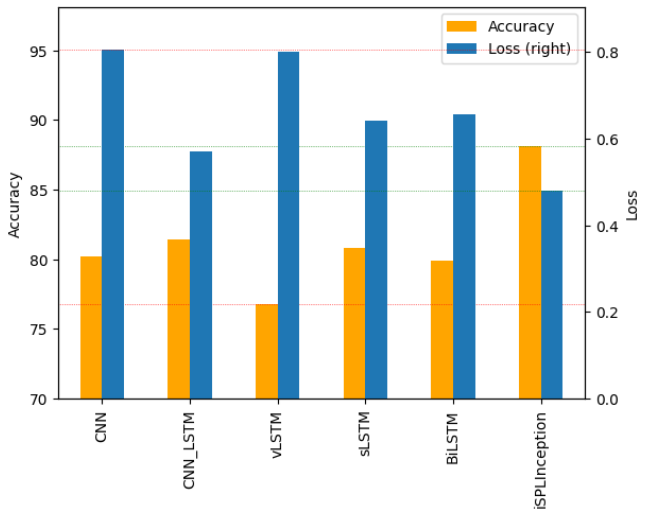

FIGURE 8. Comparing the various models on the opportunity dataset.

the models performed on the Opportunity dataset with our iSPLInception model being significantly better than the other models.

In Fig. 9, the confusion matrices from our 6 models are compared and show that the proposed model performs better

in telling the different classes apart. In most of the models, the "*Drawer*" related classes are easily mixed up because of their shared characteristics. Our iSPLInception model however performs better in telling them apart.

F. DAPHNET DATASET

Bachlin *et al.* [26] proposed the Daphnet Freezing of Gait (Daphnet) dataset. It is a dataset devised to benchmark automatic methods to recognize gait freeze from wearable acceleration sensors placed on legs and the hip. About 50% of the patients with advanced Parkinson's disease (PD) suffer from the freezing of gait (FOG), which is a sudden and transient inability to walk. It often causes falls, interferes with daily activities and significantly impairs quality of life. Because gait deficits in PD patients are often resistant to pharmacologic treatment, effective non-pharmacologic treatments are of special interest.

The goal of their study was to evaluate the concept of a wearable device that could obtain real-time gait data, process these events and provide assistance based on pre-determined specifications. They developed a real-time wearable FOG detection system that automatically provided a cueing sound when FOG is detected, and which stays until the subject resumes walking.

This wearable assistive technology was evaluated in a study with 10 PD patients. Over eight hours of data were recorded. Two hundred and thirty-seven FOG events were identified by professional physiotherapists in post-hoc video analysis. The dataset was recorded in a lab setting with an emphasis on generating many freeze events. Users performed these kinds of tasks: straight-line walking, walking with numerous turns, and finally, a more realistic activity of daily living (ADL) task, where users went into different rooms while fetching coffee, opening doors, etc.

This dataset contains 2 activities of *Freeze* and *No Freeze*. The data was sampled in fixed-width sliding windows of 3 seconds with a 50% overlap which resulted in 192 readings per window because it was recorded at 64 Hz. We used the 9 accelerometer signals, triaxial accelerometer from the ankle, upper leg, and trunk as input to the DL models.

To provide a benchmark worth reproducing and for comparing with future work, using the *datareader.py* file from [34], we split the dataset into the train, test, and validation sets based on the experiments for each subject as shown in Table 7. The disparity in percentages between the *No Freeze* and *Freeze* classes indicates a very imbalanced dataset.

In Table 8, we summarize the results from training our models on the Daphnet dataset.

Our iSPLInception model performs better on this dataset as well when compared to the other models. With an F_1 score of 94% that is 1% higher than that of the CNN, CNN-LSTM, and stacked LSTM models that each attained a 93% score. The worst model was the stacked LSTM with only 88%.

TABLE 7. Splitting the Daphnet dataset and data disparity.

Set	Subjects	Samples	No Freeze	Freeze
Train	S1-1, S1-2, S3-1 S3-2, S6-1, S6-2, S7-1, S8-1, S9-1, S10-1	7,945	91.3%	8.7%
Test	S2-1, S4-1, S5-2	2,332	92.8%	7.2%
Validation	S2-2, S3-3, S5-1	1,602	84.0%	16.0%

TABLE 8. Performance of the different models on the Daphnet dataset.

Model	Accuracy	Loss	F_1 Score	Parameters
CNN [33]	92.97	0.2768	93	628,538
CNN-LSTM [2]	92.97	0.2558	93	4,025,714
vLSTM [2]	93.22	0.2392	93	**120,397**
sLSTM [2]	87.65	0.2969	88	271,557
BiLSTM [22]	92.41	0.2479	92	238,669
iSPLInception	**93.52**	**0.2271**	**94**	1,326,726

We achieve a 93.52% model accuracy and a 0.2271 model loss which are higher than those of the stacked LSTM and CNN-LSTM networks. The proposed model parameters are only lower than those of the CNN-LSTM model like with the UCI HAR dataset.

We illustrate the training and validation accuracy and loss for the iSPLInception model in Fig. 10. We trained the proposed model and the other models for a maximum of 350 epochs with early stopping patience set to 100 epochs and used a learning rate scheduler to iteratively reduce the learning rate when the training plateaus. Fig. 11 shows how the models performed on the Daphnet dataset with our iSPLInception model showing better performance than the other models.

In Fig. 12, the confusion matrices from our 6 models are compared and show that the proposed model performs better in telling the different classes apart. In most of the models, the "*Freeze*" class is poorly classified due to the imbalance of the data. The proposed model however performs best in detecting these freeze events when compared to the other models.

G. PAMAP2 DATASET

Reiss and Stricker [25] proposed a physical activity monitoring dataset (PAMAP2) that consists of recordings from 9 participants (8 males and 1 female) instructed to carry out 18 lifestyle activities, including *household activities* (lie, sit, stand, walk, run, cycle, Nordic walk, iron, vacuum clean, rope jump, ascend and descend stairs) and a variety of *leisure activities* (watch TV, computer work, drive a car, fold laundry, clean house, play soccer). The subjects were wearing 3 inertial measurement units (IMU) that recorded the

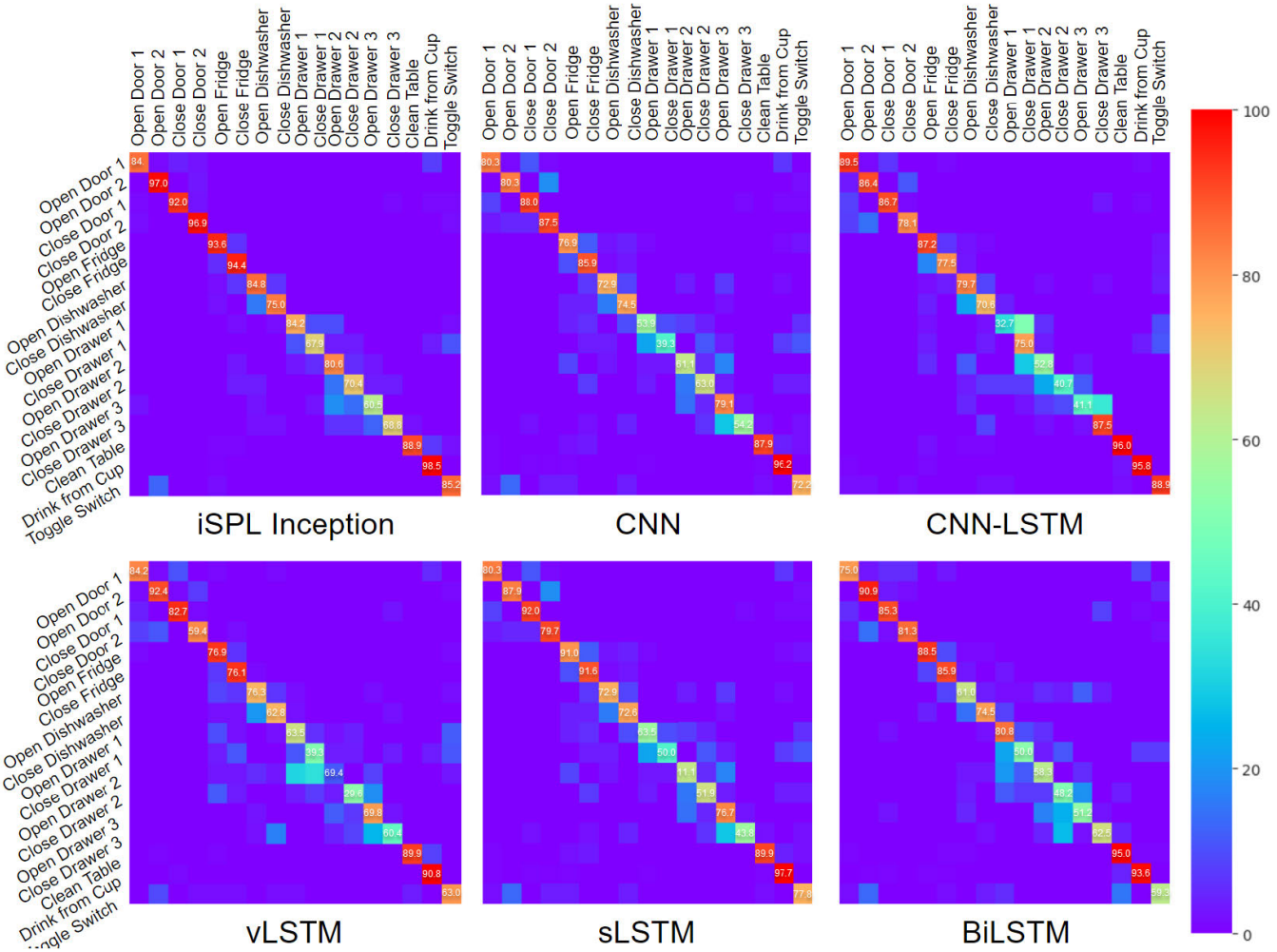

FIGURE 9. Confusion matrices for all the models on the opportunity dataset.

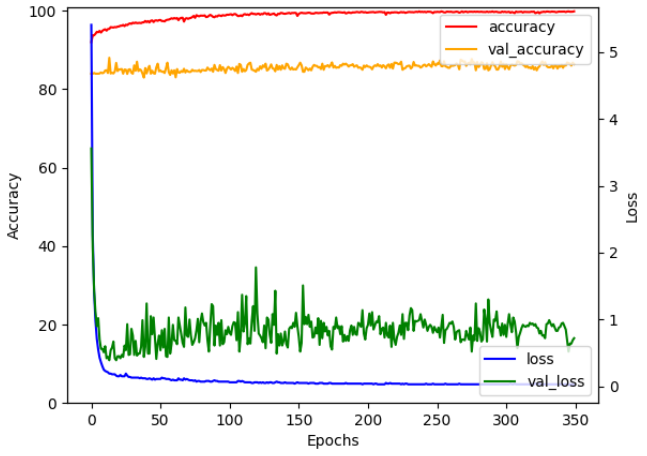

FIGURE 10. Performance of the iSPLInception model on the Daphnet dataset.

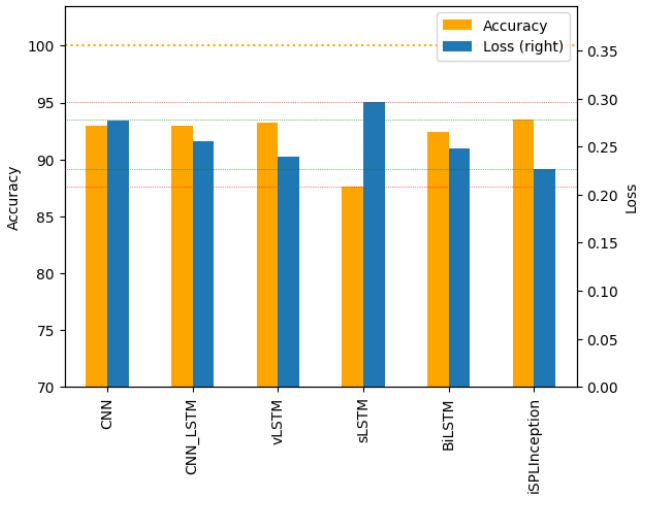

FIGURE 11. Comparing the various models on the Daphnet dataset.

accelerometer, gyroscope, magnetometer, temperature and heart rate data that were located on the hand, chest and ankle over 10 hours in total.

The PAMAP2 dataset we build has 36 dimensions. The data was sampled in fixed-width sliding windows

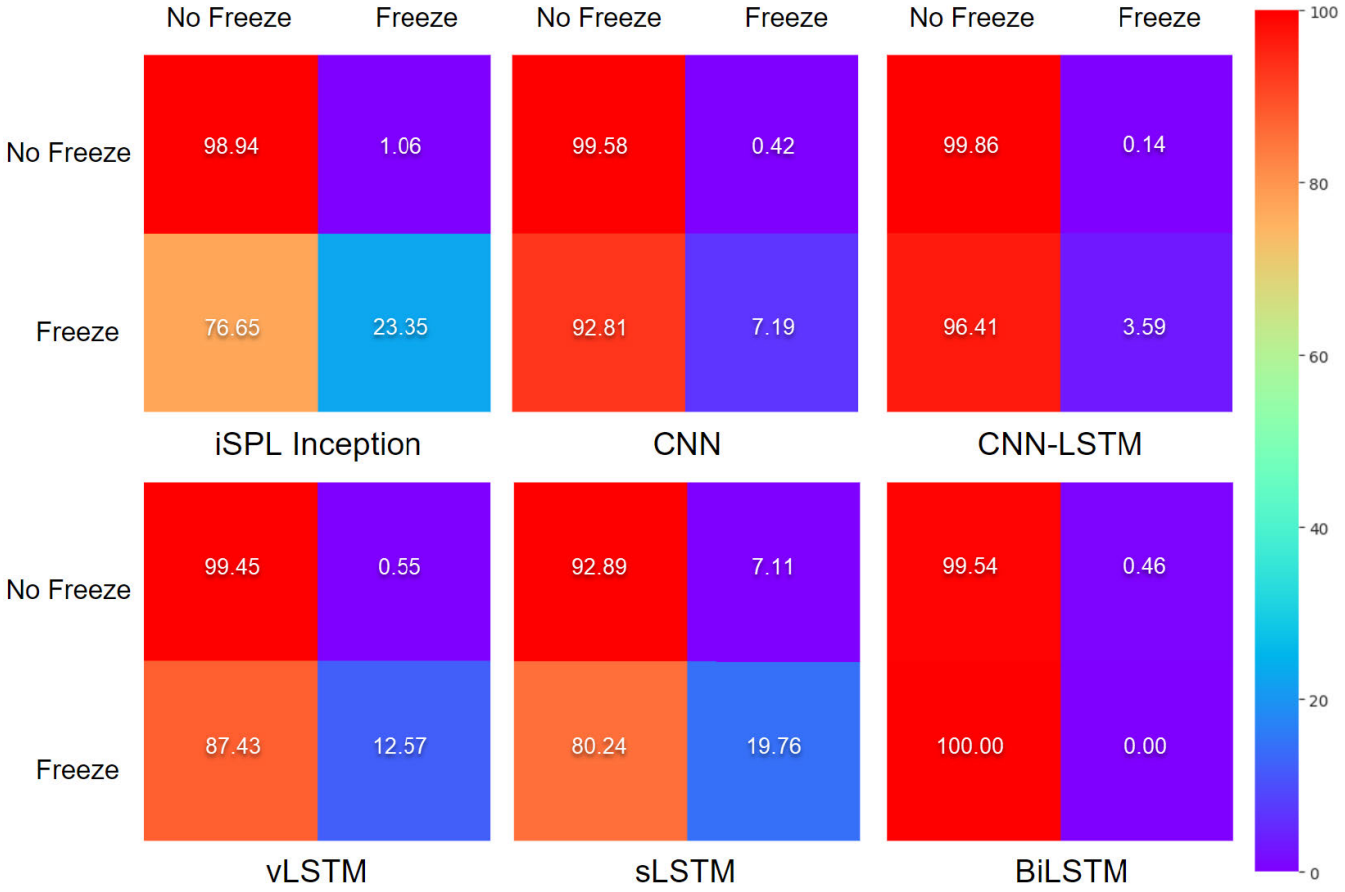

FIGURE 12. Confusion matrices for all the models on the Daphnet dataset.

TABLE 9. Splitting the PAMAP2 dataset and data disparity.

Set	Subjects	Samples	Min	Max
Train	S01, S02, S03 S04, S07, S08, S09	10,656	3.6%	12.8%
Test	S06	2,044	4.5%	15.2%
Validation	S05	1,935	4.8%	12.6%

TABLE 10. Performance of the different models on the PAMAP2 dataset.

Model	Accuracy	Loss	F_1 Score	Parameters
CNN [33]	85.78	0.6647	86	836,983
CNN-LSTM [2]	88.37	0.6174	88	4,029,323
vLSTM [2]	85.52	0.6133	86	98,635
sLSTM [2]	86.97	0.7764	87	266,687
BiLSTM [22]	86.97	0.4917	87	195,915
iSPLInception	**89.09**	**0.4322**	**89**	1,338,651

of 2.56 seconds with a 50% overlap which resulted in 256 readings per window because it was recorded at 100 Hz. We used 36 signals as input to the DL models. To provide a benchmark worth reproducing and for comparing with future work, using the *datareader.py* file from [34], we split the dataset into the train, test, and validation sets based on the experiments for each subject as shown in Table 9. The disparity in percentages between the *No Freeze* and *Freeze* classes indicates a very imbalanced dataset.

In Table 10, we summarize the results from training our models on the PAMAP2 dataset.

Our iSPLInception model performs better on this dataset as well when compared to the other models. With an F_1 score of 89% that is 1% higher than that of the CNN-LSTM model that attained an 88% score. The worst models were the CNN and vanilla LSTM with only 86%. We achieved an 89.09% model accuracy and a 0.4322 model loss which are higher than those of the stacked LSTM and CNN-LSTM networks.

We illustrate the training and validation accuracy and loss for the iSPLInception model in Fig. 13. We trained the proposed model and the other models for a maximum of 350 epochs with early stopping patience set to 100 epochs and used a learning rate scheduler to iteratively reduce the learning rate when the training plateaus. Fig. 14 compares the models' performance on the PAMAP2 dataset with our iSPLInception model showing better performance than the others.

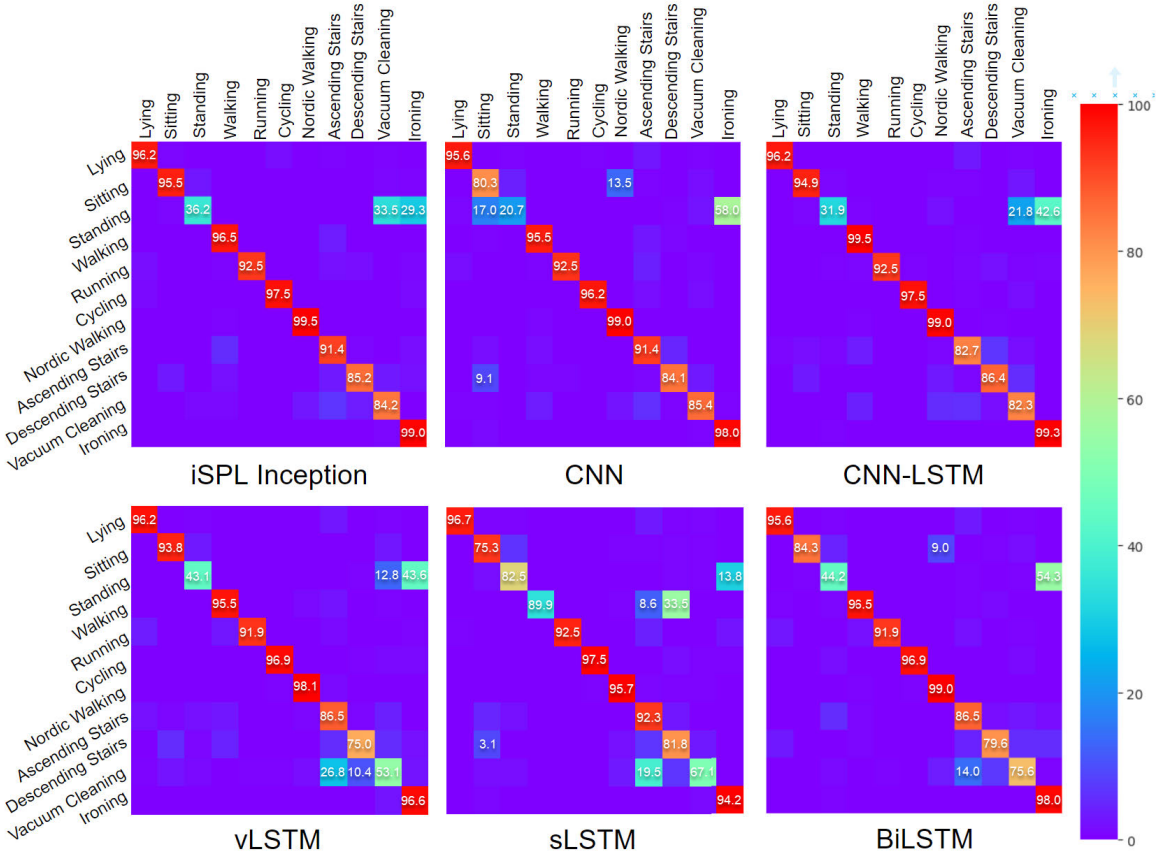

FIGURE 13. Confusion matrices for all the models on the PAMAP2 dataset.

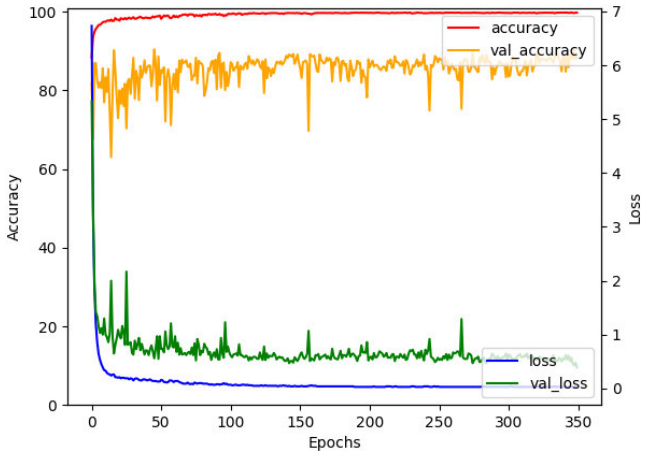

FIGURE 14. Performance of the iSPLInception model on the PAMAP2 dataset.

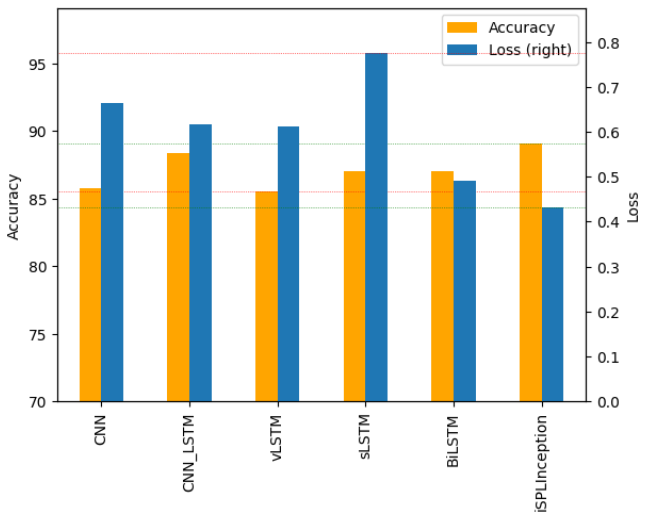

FIGURE 15. Comparing the various models on the PAMAP2 dataset.

In Fig. 15, the confusion matrices from our 6 models are compared and show that the proposed model performs better in telling the different classes apart. In all the models, the "*Standing*" class is poorly classified due to the shared characteristics with *Vacuum Cleaning* and *Ironing* since under normal circumstances, the two activities are performed while standing. The proposed model still manages to classify the classes reasonably well.

V. CONCLUSION

In this paper, we proposed the iSPLInception, a model architecture that strives to push the limits of model performance in human activity recognition. The model is based on the Inception-ResNet model and improves performance on 4 benchmarked datasets. We compared the performance of the proposed model against previous works in the realm

of HAR that include CNN, CNN-LSTM, a vanilla LSTM, and a stacked LSTM and the proposed model performs better than all these networks on the 4 datasets from the UCI machine learning repository. All the models are significantly affected by imbalances in the dataset. Another issue identified during our study was the significant change in performance when data from the different users is mixed using cross-validation. Just like in image recognition where a model will always perform poorly on previously unseen data, the HAR field suffers the same challenge. Using transfer learning has made this challenge history and our iSPLInception model is adaptable. The proposed model can be expanded to use more inception modules making it scalable with ease and having no significant detriment to performance. We believe this paper establishes a benchmark for deep learning on these four datasets for future research work and model design. For future work, we intend to use a subject independent method, like cross-validation, to evaluate each dataset. Another interesting research direction would be in using an ensemble of the best DL models as opposed to cherry-picking one.

REFERENCES

[1] R. Mutegeki and D. S. Han, "Feature-representation transfer learning for human activity recognition," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Jeju Island, South Korea, Oct. 2019, pp. 18–20.

[2] R. Mutegeki and D. S. Han, "A CNN-LSTM approach to human activity recognition," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2020, pp. 362–366.

[3] J. J. Lin, L. Mamykina, S. Lindtner, G. Delajoux, and H. B. Strub, "Fish'n'steps: Encouraging physical activity with an interactive computer game," in *Proc. Int. Conf. Ubiquitous Comput.*, 2006, pp. 261–278.

[4] K.-Y. Chen, M. Harniss, S. Patel, and K. Johnson, "Implementing technology-based embedded assessment in the home and community life of individuals aging with disabilities: A participatory research and development study," *Disab. Rehabil., Assistive Technol.*, vol. 9, no. 2, pp. 112–120, Mar. 2014.

[5] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, 2011, Art. no. 16.

[6] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 6, pp. 790–808, Nov. 2012.

[7] L. Guo, L. Wang, J. Liu, W. Zhou, and B. Lu, "HuAc: Human activity recognition using crowdsourced WiFi signals and skeleton data," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–15, Jan. 2018.

[8] J. Hong and T. Ohtsuki, "A state classification method based on space-time signal processing using SVM for wireless monitoring systems," in *Proc. IEEE 22nd Int. Symp. Pers., Indoor Mobile Radio Commun.*, Sep. 2011, pp. 2229–2233.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[10] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. 21st AAAI Conf. Artif. Intell. (AAAI)*. Palo Alto, CA, USA: AAAI Press, 2017, pp. 4278–4284.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[12] C. Xu, J. He, X. Zhang, C. Yao, and P.-H. Tseng, "Geometrical kinematic modeling on human motion using method of multi-sensor fusion," *Inf. Fusion*, vol. 41, pp. 243–254, May 2018.

[13] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga, "Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey," in *Proc. 23th Int. Conf. Archit. Comput. Syst.*, 2010, pp. 1–10.

[14] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. P. Klasnja, K. Koscher, A. LaMarca, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea, and D. Wyatt, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Comput.*, vol. 7, no. 2, pp. 32–41, Apr. 2008.

[15] T. van Kasteren and B. Krose, "Bayesian activity recognition in residence for elders," in *Proc. 3rd IET Int. Conf. Intell. Environ.*, 2007, pp. 209–212.

[16] C. Xu, D. Chai, J. He, X. Zhang, and S. Duan, "InnoHAR: A deep neural network for complex human activity recognition," *IEEE Access*, vol. 7, pp. 9893–9902, 2019.

[17] Z. Hussain, M. Sheng, and W. E. Zhang, "Different approaches for human activity recognition: A survey," *CoRR*, vol. abs/1906.05074, pp. 1–28, Jun. 2019. [Online]. Available: http://arxiv.org/abs/1906.05074.

[18] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn.*, Bruges, Belgium, 2013, p. 3.

[19] A. Jain and V. Kanhangad, "Human activity classification in smartphones using accelerometer and gyroscope sensors," *IEEE Sensors J.*, vol. 18, no. 3, pp. 1169–1177, Feb. 2018.

[20] R. Damaševičius, M. Vasiljevas, J. Šalkevičius, and M. Woźniak, "Human activity recognition in AAL environments using random projections," *Comput. Math. Methods Med.*, vol. 2016, pp. 1–17, May 2016.

[21] D. Garcia-Gonzalez, D. Rivero, E. Fernandez-Blanco, and M. R. Luaces, "A public domain dataset for real-life human activity recognition using smartphone sensors," *Sensors*, vol. 20, no. 8, p. 2200, Apr. 2020.

[22] N. T. H. Thu and D. S. Han, "Utilization of postural transitions in sensor-based human activity recognition," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Fukuoka, Japan, Feb. 2020, pp. 177–181.

[23] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. D. R. Millán, and D. Roggen, "The opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognit. Lett.*, vol. 34, no. 15, pp. 2033–2042, Nov. 2013.

[24] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Forster, G. Troster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, H. Sagha, H. Bayati, M. Creatura, and J. D. R. Millan, "Collecting complex activity datasets in highly rich networked sensor environments," in *Proc. 7th Int. Conf. Netw. Sens. Syst. (INSS)*, Kassel, Germany, Jun. 2010, pp. 233–240.

[25] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *Proc. 16th Int. Symp. Wearable Comput.*, Jun. 2012, pp. 108–109.

[26] M. Bächlin, M. Plotnik, D. Roggen, N. Giladi, J. M. Hausdorff, and G. Tröster, "A wearable system to assist walking of Parkinson's disease patients," *Methods Inf. Med.*, vol. 49, no. 1, pp. 88–95, 2010.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, Jul. 2015, pp. 448–456.

[30] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "InceptionTime: Finding AlexNet for time series classification," *Data Mining Knowl. Discovery*, vol. 34, no. 6, pp. 1936–1962, Nov. 2020.

[31] R. Gómez. (May 23, 2018). *Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and All Those Confusing Names*. Accessed: Aug. 20, 2020. [Online]. Available: https://gombru.github.io/2018/05/23/cross_entropy_loss

[32] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Conf. Oper. Syst. Design Implement. (OSDI)*. Berkeley, CA, USA: USENIX Association, 2016, pp. 265–283.

[33] D. van Kuppevelt, C. Meijer, F. Huber, A. van der Ploeg, S. Georgievska, and V. T. van Hees, "Mcfly: Automated deep learning on time series," *SoftwareX*, vol. 12, Jul. 2020, Art. no. 100548.

[34] R. Mutegeki. (Jan. 22, 2021). *iSPLInception*. Accessed: Feb. 3, 2021. [Online]. Available: https://github.com/rmutegeki/iSPLInception

[35] A. Gumaei, M. M. Hassan, A. Alelaiwi, and H. Alsalman, "A hybrid deep learning model for human activity recognition using multimodal body sensing data," *IEEE Access*, vol. 7, pp. 99152–99160, 2019.

[36] A. Gumaei, M. Al-Rakhami, H. AlSalman, S. M. M. Rahman, and A. Alamri, "DL-HAR: Deep learning-based human activity recognition framework for edge computing," *Comput., Mater. Continua*, vol. 65, no. 2, pp. 1033–1057, 2020.
[37] M. Abdel-Basset, H. Hawash, R. K. Chakrabortty, M. Ryan, M. Elhoseny, and H. Song, "ST-DeepHAR: Deep learning model for human activity recognition in IoHT applications," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4969–4979, Mar. 2021.
[38] Z. Chen, C. Jiang, S. Xiang, J. Ding, M. Wu, and X. Li, "Smartphone sensor-based human activity recognition using feature fusion and maximum full a posteriori," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 7, pp. 3992–4001, Jul. 2020.
[39] Y. Dong, X. Li, J. Dezert, M. O. Khyam, M. Noor-A-Rahim, and S. S. Ge, "Dezert–Smarandache theory-based fusion for human activity recognition in body sensor networks," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7138–7149, Nov. 2020.
[40] A. Keshavarzian, S. Sharifian, and S. Seyedin, "Modified deep residual network architecture deployed on serverless framework of IoT platform based on human activity recognition application," *Future Gener. Comput. Syst.*, vol. 101, pp. 14–28, Dec. 2019.
[41] O. Steven Eyobu and D. Han, "Feature representation and data augmentation for human activity classification based on wearable IMU sensor data using a deep LSTM neural network," *Sensors*, vol. 18, no. 9, p. 2892, Aug. 2018.

MUTEGEKI RONALD received the B.Sc. degree (Hons.) in computer science from Makerere University, Kampala, Uganda, in 2016. He is currently pursuing the M.Sc. degree in computer science and engineering with Kyungpook National University (KNU), Daegu, South Korea. In 2018, he joined the Intelligent Signal Processing Laboratory, Kyungpook National University, as a Research Student. His main research interests include sensors, localization, big-data, deep learning, and software engineering. He was a 2017 Global Korea Scholarship Scholar sponsored by the Korean Government.

ALWIN POULOSE received the B.Sc. degree in computer maintenance and electronics from the Union Christian College, Aluva, India, in 2012, the M.Sc. degree in electronics from the MES College Marampally, India, in 2014, and the M.Tech. degree in communication systems from Christ University, Bengaluru, India, in 2017. He is currently pursuing the Ph.D. degree with the School of Electronic and Electrical Engineering, Kyungpook National University, Daegu, South Korea. His research interests include indoor localization, human activity recognition, facial emotion recognition, and human behavior prediction.

DONG SEOG HAN (Senior Member, IEEE) received the B.S. degree in electronic engineering from Kyungpook National University (KNU), Daegu, South Korea, in 1987, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, Daejon, South Korea, in 1989 and 1993, respectively. From 1987 to 1996, he was with Samsung Electronics Company Ltd., where he developed the transmission systems for QAM HDTV and Grand Alliance HDTV receivers. Since 1996, he has been with the School of Electronics Engineering, KNU, as a Professor. He was a courtesy Associate Professor with the Department of Electrical and Computer Engineering, University of Florida, in 2004. He was the Director of the Center of Digital TV and Broadcasting, Institute for Information Technology Advancement, from 2006 to 2008. His main research interests include intelligent signal processing and autonomous vehicles.