applied sciences

MDPI

Article

# Computation of Psycho-Acoustic Annoyance Using Deep Neural Networks

**Jesus Lopez-Ballester *,†, Adolfo Pastor-Aparicio †, Jaume Segura-Garcia †, Santiago Felici-Castell † and Maximo Cobos †**

Computer Science Department, Escola Tècnica Superior d'Enginyeria, Universitat de València, 46100 Burjassot, Spain
* Correspondence: jesus.lopez-ballester@uv.es
† These authors contributed equally to this work.

Received: 28 June 2019; Accepted: 1 August 2019; Published: 2 August 2019


**Abstract:** Psycho-acoustic parameters have been extensively used to evaluate the discomfort or pleasure produced by the sounds in our environment. In this context, wireless acoustic sensor networks (WASNs) can be an interesting solution for monitoring subjective annoyance in certain soundscapes, since they can be used to register the evolution of such parameters in time and space. Unfortunately, the calculation of the psycho-acoustic parameters involved in common annoyance models implies a significant computational cost, and makes difficult the acquisition and transmission of these parameters at the nodes. As a result, monitoring psycho-acoustic annoyance becomes an expensive and inefficient task. This paper proposes the use of a deep convolutional neural network (CNN) trained on a large urban sound dataset capable of efficiently predicting psycho-acoustic annoyance from raw audio signals continuously. We evaluate the proposed regression model and compare the resulting computation times with the ones obtained by the conventional direct calculation approach. The results confirm that the proposed model based on CNN achieves high precision in predicting psycho-acoustic annoyance, predicting annoyance values with an average quadratic error of around 3%. It also achieves a very significant reduction in processing time, which is up to 300 times faster than direct calculation, making CNN designed a clear exponent to work in IoT devices.

**Keywords:** convolutional neural networks; psycho-acoustic parameters; subjective annoyance; wireless acoustic sensor networks; Zwicker model

## 1. Introduction

In recent years, Wireless acoustic sensor networks (WASN) have received increasing attention in the field of acoustic signal processing and machine learning research. Many novel approaches for solving classical problems such as acoustic source localization [1,2], acoustic event detection and classification, or environmental noise evaluation [3] have been revisited assuming different scenarios that consider wireless acoustic nodes [4]. Additionally, in recent decades, many studies have been carried out to measure noise pollution, most of them using the acquired sound pressure level over time on the basis of a standard [5]. An example are the traffic noise control maps developed in Beijing [6] or London [7] to describe spatially the environmental noise of such cities. These measurements give us information about the weighted amplitude of the noise, i.e., "A-weighted" scale of SPL in dB(A). However, although they are somehow correlated with other spectral metrics, they provide little information about the mechanisms of action of the environmental noise on the people's mood and the subjective annoyance produced by these sounds [8]. To establish how annoying is the sound in a specific environment, other magnitudes in addition to the sound pressure level are needed. This is where the psycho-acoustic parameters, such as loudness, sharpness, roughness or fluctuation strength [9] come in. These parameters provide a better approximation of the annoyance caused by the different types of sounds perceived by

human subjects, allowing a more precise analysis of the acoustic environment. An example of the use of these parameters using an artificial neural network for classification can be shown in [10], where the psycho-acoustic parameters are used together with subjective surveys to classify the degree of annoyance or wellness of different recordings made in crowded areas within the city. The application of the soundscape description, as defined in ISO 12913-1 to a Smart City [11], involves a real-time implementation of these psycho-acoustic parameters. Traditionally, Zwicker's annoyance model [9] has been used to describe the degree of subjective nuisance produced by a noise-producing device or even in an environment by means of the determination of different psycho-acoustic parameters, namely loudness, sharpness, roughness, and fluctuation strength, to return a sound nuisance indicator. A problem arising from the use of such parameters is the significant computing time required, which presently does not allow calculation of them in real time. To lower the computing time for IoT-based systems, different approaches have been developed [8,12,13], but no one was efficient enough to develop an autonomous real-time application for a WASN.

In this paper, we propose the use of a regression model based on convolutional neural networks (CNNs) to be used as a tool to predict the total psycho-acoustic annoyance (PA) indicator from raw audio signals. To this end, we have used a training database based on the taxonomy established by ISO 12913-2 [14] for urban sounds, using the UrbanSound8K dataset [15] expanded with several recordings of our own made in different urban spaces and situations. The trained CNN-based model is shown to be around 300 times faster than the direct computation approach, providing as well significant accuracy on the test dataset.

## 2. Psycho-Acoustic Annoyance Model

The evaluation of psycho-acoustic annoyance over a considerably large space (e.g., a block of buildings or a city neighborhood), can be performed by means of a WASN. Due to privacy and network speed reasons, it is essential that only one general nuisance parameter is transmitted at a certain rate, so that nuisance calculations must be performed in real time at each node to optimize the system's performance. Traditionally, psycho-acoustic annoyance has been evaluated considering Zwicker's model [9], which is the chosen model used throughout this work. This section describes the parameters involved in the computation of psycho-acoustic annoyance using Zwicker's model.

### 2.1. Zwicker's Psycho-Acoustic Annoyance Model

Psycho-acoustic parameters allow us to quantify the degree of subjective discomfort (or pleasantness), in terms of objective metrics that certain sounds produce in people. The Zwicker's annoyance model [9] uses 4 psycho-acoustic parameters: Loudness ($L$), Sharpness ($S$), Fluctuation Strength ($F$) and Roughness ($R$) to calculate a general annoyance parameter ($PA$).

This model is based on the anatomy of the human ear, which behaves like a bank of filters implemented in the cochlea, the sensory organ of hearing located within the inner ear. Thus, the frequency spectrum of psycho-acoustic metrics is analyzed in terms of *critical bands* [16] with a frequency bandwidth matching the response of the mentioned auditory filters. Sound stimuli that are very close to each other in terms of frequency are combined in the human ear in the same critical band. Serializing these critical bands, we create two different frequency scales, the Bark scale and the ERB scale, called *critical band rate scales*, one measured in *Barks* and the other one measured in *Equivalent Rectangular Bandwidth* (*ERBs*) [9].

Table 1 summarizes the parameterized formulas used for calculating the psycho-acoustic metrics [9] involved in Zwicker's model. In the following, we briefly describe the parameters involved in the computation. For further details on each one, the reader is referred to [9].

- Psycho-acoustic Annoyance ($PA$) is a perceptual attribute that allows an objective quantification of noise annoyance from the physical characteristics of the signal, based on the mean values of $L$, $S$, $R$, and $F$.

- Loudness ($L$) is a perceptual measure of the effect of the energy content of sound on the ear (intensity sensation), measured in *Sones* using a linear scale. It is standardized in ISO 532B. The process used to calculate $L$ is based on the Specific Loudness ($L'(z)$ or $L$ contribution for the $z$-th critical band, measured in *Sone/Bark*. The total $L$ is the result of accumulating all contributions across the different bands, weighted by their specific bandwidth $\Delta z$.
- Sharpness ($S$) is a value of sensory human perception of unpleasantness in sounds that is caused by high frequency components. It is measured in *Aures* in a linear scale.
- Roughness ($R$) describes the perception of the sound fluctuation even when $L$ or $L_{eq,T}$ (i.e., the equivalent continuous sound level) remains unchanged. It analyzes the effects with different degrees of frequency modulations (around 70 Hz) in each critical band. The basic unit for $R$ is *Asper*. For each *ERB*, $g(z)$ is an arbitrary weighting function, $m$ is the modulation depth of each *ERB* and $k$ is the cross-correlation between the envelopes of the *ERB* with indexes $i$ and $i-2$.
- Fluctuation Strength ($F$) describes how strongly or weakly sounds fluctuate. It depends on the frequency and depth of the $L$ fluctuations, around 4 Hz in each *ERB*. It is measured in *Vacils*.

**Table 1.** Numerical expressions for $PA$, $L$, $S$, $R$, and $F$.

| |
|---|
| $PA = L\left(1+\sqrt{\left(\frac{(S-1.75)log(L+10)}{4}\right)^2+\left(\frac{2.18(0.4F+0.6R)}{L^{0.4}}\right)^2}\right)$ |
| $L = \sum_{z=0}^{28Bark} L'(z)\cdot\Delta z$ |
| $S = C_S\cdot\frac{\sum_{z=0}^{28Bark} L'(z)\cdot e^{0.171\cdot z}\cdot z\cdot\Delta z}{L}$ |
| $R = C_R\cdot\sum_{i=0.5}^{33\,ERB}(g(z_i)\cdot m_i\cdot k_{i-2}\cdot k_i)^2$ |
| $F = C_F\cdot\sum_{i=0.5}^{33\,ERB}(g(z_i)\cdot m_i\cdot k_{i-2}\cdot k_i)^2$ |

The terms $C_S$, $C_R$ and $C_F$ are calibration constants.

### 2.2. Signal Processing and Computing Time

To calculate the nuisance model we designed four different scripts aimed at computing each of the psycho-acoustic parameters ($L$, $S$, $R$, and $F$) from an input audio signal and one to calculate the total annoyance PA from the rest of parameters. As we need to simplify the computation, the study is oriented towards an acoustic sensor network framework in which we use a single-channel audio input per node. A sampling frequency of 16 kHz is used to reduce the number of samples to be processed. Then, annoyance is analyzed over frequencies up to 8 kHz. As performed in many other studies [8,17,18] and as suggested in Zwicker's book [9], the input signal is enframed with typical window sizes of 80 to 200 ms with 50% overlap. This applies to $L$, $S$, and $R$, but $F$ needs longer windows, as it takes into account modulation frequencies of about 4 Hz and performs better with window sizes in the range going from 500 ms to 1 s. In this work, we considered windows with a length of 1 s.

To perform a descriptive measurement of the required computing time of psycho-acoustic parameters and the general annoyance parameter PA, we have used three different platforms: two personal computers and a RaspberryPi-3B. The times of the personal computers were used to obtain an average of the processing time in common desktop equipment. To get an idea of the average calculation time on a normal desktop PC, we have evaluated 1000 audios on each of the 2 computers, obtaining an average of 1000 times on each. In Table 2 you can see an average of these 2 average times, in order to have a baseline of a generic PC performance calculating the general psycho-acoustic annoyance indicator, PA, and compare it with an IoT device, such as the Raspberry Pi. The Raspberry Pi platform was used to estimate the average computing time in a real node, as it is a device widely used in WASN applications [8,13,19]. The specifications are:

- PC-1: Intel(R) Core i7-7700CPU 3.6 Ghz x64; 16 GB RAM; NVIDIA GTX 1060 6 GB.

- PC-2: 2 × Intel(R) Xenon(R)Silver 2.2 Ghz x64; 96 GB RAM; NVIDIA GTX 1080 8 GB.
- Raspberry Pi 3B: CPU + GPU: Broadcom BCM2837 Cortex-A53 (ARMv8) x64 1.2 GHz; 1 GB RAM.

**Table 2.** PA computing time comparison in seconds.

| | L | S | R | F | PA |
|---|---|---|---|---|---|
| Avg. t PC | 0.0561 | 0.0001 | 0.5152 | 0.6429 | 1.2143 |
| RPi3B t | 0.0610 | 0.0001 | 0.8520 | 0.7423 | 1.6554 |

We used 1000 audio clips of one second duration to carry out the computing tests. These were taken randomly from our database (discussed later), measuring the computation time of each parameter in all platforms. Table 2 shows the results of the computing times. In general, *S* is the fastest as it is taken directly from *L* as described in Table 1. The calculation of *R* and *F* are the slowest ones. Looking at the results of a Raspberry Pi 3B, it can be observed that times are well above 1 s, which does not allow real-time processing. Note also that the resulting times would considerably increase for higher sampling rates. Although the use of two desktop PCs with dedicated GPUs can be confusing, in no case has the GPU been used to perform direct calculations or CNN predictions. Especially so that the conditions of the test were the same regarding the Raspberry Pi. Therefore, both the direct calculations and the predictions have been made in the CPU and only affect the characteristics of the CPU.

A more extensive study on the calculation of psycho-acoustic parameters in a WASN using Raspberry Pi is provided in [19]. Although more powerful platforms could be used for this purpose, the deployment cost would also increase significantly [20]. Thus, these results motivate the proposal of a different computational approach capable of predicting faster PA values. The following section describes our proposal based on convolutional neural networks.

## 3. Materials and Methods

As discussed in the previous section, an alternative tool to the direct computation of the psycho-acoustic metrics is needed to perform real-time PA evaluation within an IoT node. In recent years, deep neural networks (DNNs) have been extensively used to solve a wide range of problems related to audio signal processing, such as audio event detection [21–23], source separation [24] or source localization [25]. In this context, CNNs have been shown to be a powerful tool for many audio-related tasks, with internal layers that are able to learn optimized features capturing those signal properties that are relevant to solve the task at hand. In this work, we propose to train a CNN to solve a regression problem, where raw audio windows are provided as input, receiving as an output the predicted PA value. To this end, the PA ground-truth values used for training are obtained by direct computation, as described in Section 2 .

The data processing, the calculation of the psycho-acoustic parameters discussed above and the development of the CNN described in this section have been performed in *Matlab R2018b* and *R2019a*, making use of the functions included in the toolboxes: *DSP System Toolbox, Deep Learning Toolbox and Matlab Coder*.

### *3.1. Datasets*

The considered audio signals used in this work belong to the UrbanSound8K database [15], consisting of 8000 tagged audios following a taxonomy similar to the one described in ISO 12913-2 [14] for urban sounds. Please note that in our case we are not especially interested in the labels, as we do not intend to recognize audio classes. All the files with a length greater than 1 s were considered and split to obtain a total of 59,000 one-second audio segments. The files were originally recorded with different sampling frequencies from 8 kHz to 48 kHz. This database was extended with 1150 s of recordings in different cities and areas, resampling all the final segments to 16 kHz. Finally, the whole database with 60,150 audio segments was divided into three datasets for training, validation,

and test. To this end, we extracted randomly 1000 audios to generate the test dataset. The remaining 59,150 signals were used for training (80%, 47,320 signals) and validation (20%, 11,830 signals). For all the audio segments, we extracted by direct computation their corresponding PA value, to use the computed values as ground-truth annoyance during the training and validation of the system. Since calibration information is missing, we assumed a standard mapping to SPL as typically performed in audio coding. Bearing in mind that PA values range from 0 to 100, proportional to unpleasant to extremely annoying sounds, it can be observed in the histogram shown in Figure 1 that most audio segments in the database have PA values in the range going from 0 to 50, so higher accuracy is expected to be achieved by system in this range.

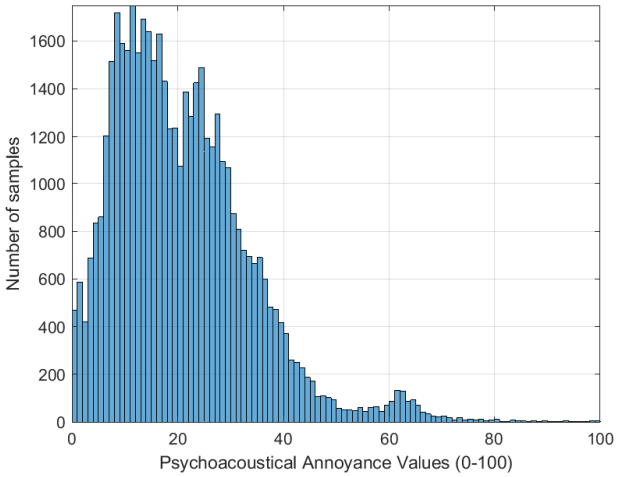


**Figure 1.** Histogram of PA values in the database.

### *3.2. Regression CNN Design and Training*

The scheme summarizing the CNN architecture used in this work is shown in Figure 2. After the input layer, basically, the neural network is based on 4 *Convolutional* stages. Each of these stages consists of a block, formed by 4 layers:

- **Convolutional layer**, that applies sliding convolutional filters to the input.
- **Batch Normalization layer**, that normalizes each input channel across a mini-batch to speed up the training process.
- **Rectified Linear Unit (ReLU) layer**, which sets to zero any negative input value.
- **Max-pool layer**, that performs a down-sampling of the input by dividing the input into same size pooling regions, and computing the maximum of each region.

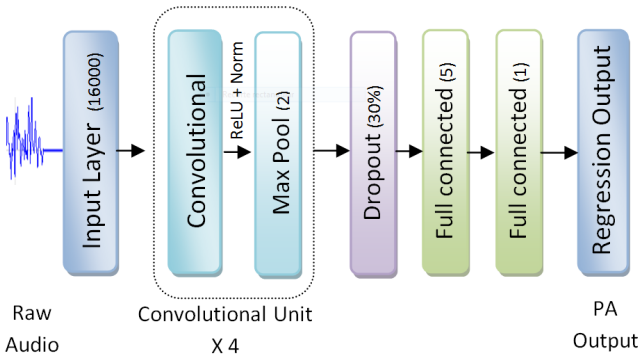


**Figure 2.** Regression CNN Design.

This convolutional unit is repeated 4 times in our design, as we have said and can be seen summarized in Figure 2, marked as "Convolutional Unit". In Table 3 these 4 convolutional units are shown separated by horizontal lines and you can see in detail the layers that form them. This scheme of our convolutional unit is the one that worked best when adding new convolutional layers to the overall design. This scheme allows us to progressively reduce the number of data and converge towards a very accurate PA prediction. After these 4 convolutional units a *Dropout* layer is placed with dropout probability of 0.3, followed by 2 *Fully connected* layers that reduce the output to 5 elements and then to 1, which reaches the final *Output Regression* layer that predicts the corresponding PA value. The full description of the parameters corresponding to each layer is shown in Table 3.

**Table 3.** CNN layer description.

| Layer | Size | Filters | Stride |
|---|---|---|---|
| Input | 16,000 × 1 | | |
| Convolutional S1 | 512 × 1 | 10 | 10 |
| Batch Norm. S1 | | | |
| ReLU. S1 | | | |
| Max Pool S1 | 2 × 1 | | 2 |
| Convolutional S2 | 256 × 1 | 20 | 5 |
| Batch Norm. S2 | | | |
| ReLU. S2 | | | |
| Max Pool S2 | 2 × 1 | | 2 |
| Convolutional S3 | 128 × 1 | 40 | 2 |
| Batch Norm. S3 | | | |
| ReLU. S3 | | | |
| Max Pool S3 | 2 × 1 | | 2 |
| Convolutional S4 | 64 × 1 | 60 | 2 |
| Batch Norm. S4 | | | |
| ReLU. S4 | | | |
| Max Pool S4 | 2 × 1 | | 2 |
| Convolutional S5 | 32 × 1 | 80 | 1 |
| Batch Normalization S5 | | | |
| ReLU | | | |
| Max Pool S5 | 2 × 1 | | 2 |
| Dropout 30% | | | |
| Fully Connected | 1 × 5 | | |
| Regression Output | 1 × 5 | | |

In the column "Size" of Table 3 we describe the input dimensions of each layer, so, when we indicate that the "Input layer" has a size of 16,000 × 1 it means that its input will be a vector of 16,000 samples, which in our case refers to 1 s of audio (1 channel) sampled at 16 KHz. Each layer has an input and an output and the size of the output of one layer corresponds to the input size of the next layer. A convolutional layer applies sliding convolutional filters to the input of the layer, so that its "Size" column represents the size of each filter implemented. Taking as an example the first convolutional layer, it is formed by filters of size 512 × 1 since its entry will be a vector also (16,000 × 1). Specifically, it is formed by 10 filters of size 512 × 1, as it is indicated in the column "Filters".

Observing the Convolutional layers and the Max-Pool layers, we see that they have data in the column "Stride", this indicates us the number of elements that we move the filters before applying them again. Therefore, simplifying, we apply an A × B size filter, we get the result and we move the filter C samples before applying it again, and we repeat this operation until we reach the end of the layer input vector. Therefore, in a Max-Pool layer with size 2 × 1 and a "Stride" of 2, imagining that we move from left to right along the input data vector, we would take the largest of 2 elements, we

would move 2 samples to the right, we would take again the largest of the following 2 elements, and so on.

The training process has been carried out using the desktop computer listed as PC-2 in the Section 2.2. It is the one case of all this study in which the GPU has been used, but as we have already mentioned, only for the training of the neural network. In no case has any dedicated GPU been used to make calculations or predictions.

The training options used were as follows:

- *Solver*: SGDM
- *Momentum*: 0.9000
- *InitialLearnRate*: $1.0000 \times 10^{-3}$
- *L2Regularization*: $1.0000 \times 10^{-4}$
- *GradientThresholdMethod*: L2 Norm
- *MaxEpochs*: 70
- *MiniBatchSize*: 128
- *ValidationData*: 11,830 elements
- *ValidationFrequency*: 1107
- *Shuffle*: Every epoch

We have tried different training options to evaluate the design of our CNN, changing all the parameters mentioned above. Testing with 2, 3, and 4 convolutional stages, the options chosen are those that have yielded the best root mean square error (RMSE) results. Maintaining these options, therefore, we have tested different CNN designs with more or fewer layers and so far the configuration described in this article is the least error in prediction we get.

If we take the PA prediction, as this usually takes values in the range of 0 to 100, so observing the error made in the prediction of PA, gives us a direct idea of the percentage of error committed. Later we will discuss the error in prediction made in more detail, but we can advance by looking at Table 4 RMSE of PA prediction by using different number of stages or convolutional units. The configuration with 4 convolutional stages is a trade-off between complexity of several layers, the lowest RMSE, training and prediction times. An example of these RMSE values are shown in Table 4. It is worth mentioning that the results shown in Table 4 have been obtained by measuring in the validation dataset (11,830 signals), so the RMSE can be reduced by using the test dataset (1000 signals), as we will see later.

**Table 4.** Root mean square error (RMSE) with different number of convolutional stages.

| Nº Conv. Stages | PA Prediction RMSE |
|---|---|
| 2 stages | 12.981 |
| 3 stages | 4.512 |
| 4 stages | 3.039 |

## 4. Evaluation and Results

After carrying out the training process of the network, the system was evaluated by using the validation dataset and through an independent test dataset (which has not been part of the training process). Considering such datasets, we measured the RMSE to evaluate the accuracy of the predictions. Also, the required time to make predictions was evaluated as well to test the speed of the CNN. The results are described in this section.

### *4.1. Training Results and Accuracy Test*

The training process produces an RMSE value of 3.0393 by comparing the directly calculated PA of the validation dataset with the predicted PA using the trained CNN. Taking into account a

PA range from 0 to 100, this is the best value achieved after testing several CNN architectures and taking into account that PA values are considerably unbalanced (Figure 1). Similarly, we used the 1000 audio clips of the test dataset obtaining an RMSE equal to 1.7672. The improvement of the RMSE over the test dataset (1000 signals) is probably due to the fact that it is much smaller than the validation dataset (11,830 signals) and contains fewer PA values located at the extremes of the range, where most errors occur.

In Figure 3 (left) we can see a comparison between the calculated PA values (ground-truth) represented by red circles and the ones predicted by the CNN, represented by black crosses. In the right part of the same Figure 3 we can see the prediction error made by CNN. The same order of the samples is maintained so we have marked the same areas as in the left side of the figure, where the error tends to be greater due to the fact that we had fewer audio signals with PA values higher than 45 to train the CNN.

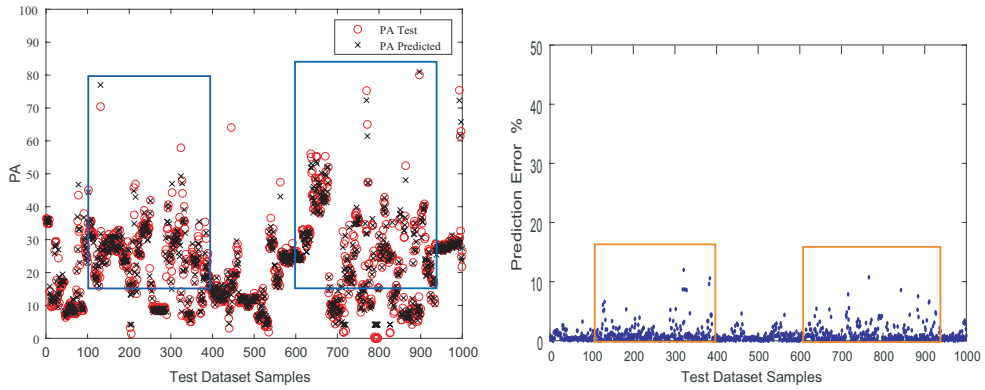


**Figure 3.** PA calculated vs. CNN prediction (**left**), CNN prediction error (**right**) using test dataset. Areas marked with high PA values where the highest prediction errors are located.

A scatter plot of the predicted PA values versus the calculated ground-truth values is shown in Figure 4, where we can observe more clearly the deviations of the predicted values across the PA range going from 0 to 100. In the figure we have marked with circles the values that move more than 10% away from the ideal straight line (in red). We can see that they are only 5 predictions of the 1000 made over the signals of the test dataset and that they are placed in values superior to 50 of PA, due as we have commented previously to the lack of audio samples with values of high PA in our database. As expected, the error tends to be higher for those audio segments with extremely high PA values but, overall, the CNN model provides PA predictions with very good accuracy in most cases.

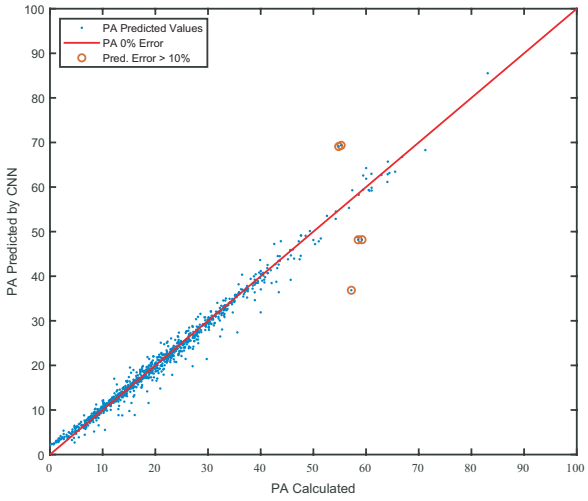


**Figure 4.** PA Calculated vs. PA Predicted by CNN. Predicted PA values with Error > 10% marked.

### *4.2. Direct Calculation vs. CNN Prediction in Terms of Computation Time*

This section discusses the differences in computation time required by PA direct calculation and CNN PA prediction. To this end, we measured the time taken by the algorithm to calculate the psycho-acoustic parameters and psycho-acoustic annoyance PA, and the time taken by the neural network to predict the PA value, using the computers described in Section 2.2 in both cases. The results are shown in Figure 5.

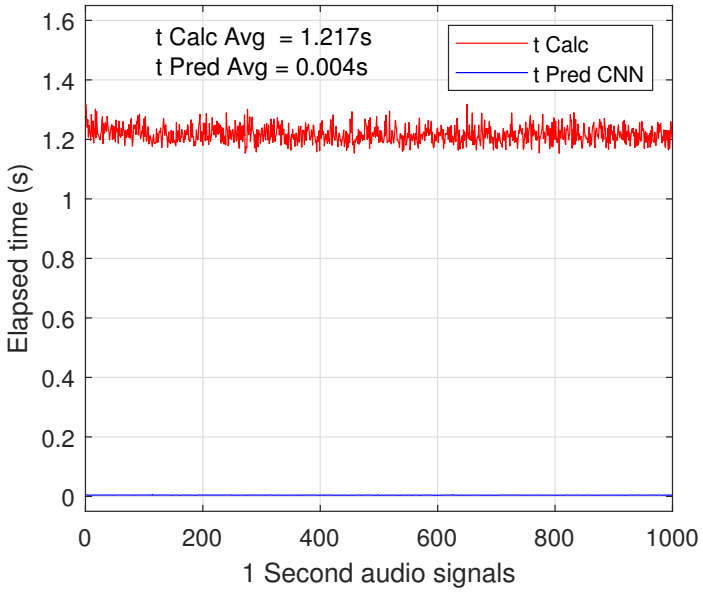


**Figure 5.** Elapsed time in direct calculation and elapsed time in prediction.

The time used by CNN to predict the PA value (blue) from raw audio signals is significantly smaller than the one obtained by direct calculation (red). The average time for CNN prediction is 0.004 s while for direct calculation is 1.217 s. Thus, a speedup higher than 300 is achieved by the proposed CNN-based system. It can also be observed that the times obtained from the CNN are almost constant while the times obtained by direct calculation depend on the complexity of the input sound. All these results confirm that the proposed CNN model is a good candidate for IoT-based

implementations, where the PA analysis can be comfortably performed in real time with a computation time much smaller than 1 s.

## 5. Conclusions

In this paper, we proposed a convolutional neural network to estimate the psycho-acoustic annoyance from raw audio signals more efficiently than by using direct calculation. The proposed CNN was shown to be very accurate in predicting the PA value of an input audio signal, quantifying discomfort according to the classical Zwicker's annoyance model. To train such model, we used a large dataset of urban sounds, using the computed annoyance over 1 s segments as ground-truth values for training and validation. Despite having a significantly unbalanced dataset, the resulting model has been confirmed to predict with a very small error the PA values in the range going from 0 to 50 PA units, and moderate errors in those audio segments presenting extremely high PA values. On the other hand, we also compared the computing time required by the proposed model and the one obtained by means of direct calculation, obtaining a speedup higher than 300. We have demonstrated the effectiveness of using deep neural networks to estimate PA in IoT devices with limited resources, performing computations in the same node and implementing a smart WASN more easily and efficiently.

**Author Contributions:** The authors contributed as follows: conceptualization, J.Lopez.; methodology, J.Lopez and A.Pastor.; software, J.Lopez and A.Pastor.; validation, J.Lopez, A.Pastor and J.Segura.; formal analysis, J.Lopez and M.Cobos; investigation, J.Lopez.; resources, S.Felici, M.Cobos and J.Segura.; data curation, J.Lopez; writing–original draft preparation, J.Lopez.; writing–review and editing, J.Lopez, J.Segura and S.Felici; supervision, M.Cobos, J.Segura and S.Felici; project administration, J.Segura, S.Felici and M.Cobos; funding acquisition, J.Segura, S.Felici and M.Cobos.

**Funding:** This work has been funded by the Spanish Ministry of Economy and Competitiveness and co-funded with European Regional Development Fund (FEDER) under the project grants with reference BIA2016-76957-C3-1-R and RTI2018-097045-B-C21.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Cobos, M.; Perez-Solano, J.; Felici-Castell, S.; Segura Garcia, J.; Navarro, J.M. Cumulative-Sum-Based Localization of Sound Events in Low-Cost Wireless Acoustic Sensor Networks. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2014**, *22*, 1792–1802. [CrossRef]
2. Cobos, M.; Antonacci, F.; Alexandridis, A.; Mouchtaris, A.; Lee, B. A Survey of Sound Source Localization Methods in Wireless Acoustic Sensor Networks. *Wirel. Commun. Mob. Comput.* **2017**, *2017*, 3956282. [CrossRef]
3. Noriega-Linares, J.E.; Rodriguez-Mayol, A.; Cobos, M.; Segura Garcia, J.; Felici-Castell, S.; Navarro, J.M. A Wireless Acoustic Array System for Binaural Loudness Evaluation in Cities. *IEEE Sens. J.* **2017**, *17*, 7043–7052. [CrossRef]
4. Cobos, M.; Perez-Solano, J.J.; Berger, L.T. Acoustic-based technologies for ambient assisted living. In *Introduction to Smart eHealth and eCare Technologies*; Sari Merilampi, A.S., Ed.; Taylor & Francis Group: Boca Raton, FL, USA, 2016; Chapter 9, pp. 159–177.
5. International Organization for Standardization (ISO). *ISO 1996-1: 2016, Acoustics-Description, Measurement and Assessment of Environmental Noise—Part 1: Basic Quantities and Assessment Procedures*; Technical Report; ISO: Geneva, Switzerland, 2016.
6. Li, B.; Tao, S.; Dawson, R. Evaluation and analysis of traffic noise from the main urban roads in Beijing. *Appl. Acoust.* **2002**, *63*, 1137–1142. [CrossRef]
7. NoiseMap Ltd. Noise Map, Environmental Noise Mapping Software. 2018. Available online: http://www.londonnoisemap.com (accessed on 19 June 2019).
8. Segura-Garcia, J.; Felici-Castell, S.; Perez-Solano, J.J.; Cobos, M.; Navarro, J.M. Low-Cost Alternatives for Urban Noise Nuisance Monitoring Using Wireless Sensor Networks. *IEEE Sens. J.* **2015**, *15*, 836–844. [CrossRef]

9. Fastl, H.; Zwicker, E. *Psychoacoustics: Facts and Models*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 327–329.
10. Maffei, L.; Masullo, M.; Toma, R.A.; Ciaburro, G.; Firat, H.B. Awaking the awareness of the movida noise on residents: Measurements, experiments and modelling. In Proceedings of the 48th Inter-Noise, Madrid, Spain, 16–19 June 2019.
11. International Organization for Standardization (ISO). *ISO 12913-1: 2014, Acoustics-Soundscape—Part 1: Definition and Conceptual Framework*; Technical Report; ISO: Geneva, Switzerland, 2014.
12. Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of Things for Smart Cities. *IEEE Internet Things J.* **2014**, *1*, 22–32. [CrossRef]
13. Segura-Garcia, J.; Perez-Solano, J.J.; Cobos, M.; Navarro, E.; Felici-Castell, S.; Soriano, A.; Montes, F. Spatial Statistical Analysis of Urban Noise Data from a WASN Gathered by an IoT System: Application to a Small City. *Appl. Sci.* **2016**, *6*, 380. [CrossRef]
14. International Organization for Standardization (ISO). *ISO/TS 12913-2: 2018, Acoustics-Soundscape—Part 2: Data Collection and Reporting Requirements*; Technical Report; ISO: Geneva, Switzerland, 2018.
15. Salamon, J.; Jacoby, C.; Bello, J.P. A Dataset and Taxonomy for Urban Sound Research. In Proceedings of the 22nd ACM International Conference on Multimedia (MM '14), Orlando, FL, USA, 3–7 November 2014; ACM: New York, NY, USA, 2014; pp. 1041–1044. [CrossRef]
16. Gelfand, S.A. *Hearing: An Introduction to Psychological and Physiological Acoustics*; CRC Press: Boca-Raton, FL, USA, 2017.
17. Terhardt, E.; Stoll, G.; Seewann, M. Algorithm for extraction of pitch and pitch salience from complex tonal signals. *J. Acoust. Soc. Am.* **1982**, *71*, 679–688. [CrossRef]
18. Lingsong, H.; Crocker, M.J.; Ran, Z. FFT based complex critical band filter bank and time-varying loudness, fluctuation strength and roughness. In Proceedings of the International Congress on Sound and Vibration 2007 (ICSV14), Cairns, Australia, 9–12 July 2007.
19. Pastor-Aparicio, A.; Lopez-Ballester, J.; Segura-Garcia, J.; Felici-Castell, S.; Cobos, M.; Fayos-Jordan, R.; Perez-Solano, J. Real time implementation for psycho-acoustic annoyance monitoring on wireless acoustic sensor networks. In Proceedings of the 48th Inter-Noise, Madrid, Spain, 16–19 June 2019.
20. Belloch, J.A.; Badía, J.M.; Igual, F.D.; Cobos, M. Practical Considerations for Acoustic Source Localization in the IoT Era: Platforms, Energy Efficiency and Performance. *IEEE Internet Things J.* **2019**, *6*, 5068–5079. [CrossRef]
21. Aytar, Y.; Vondrick, C.; Torralba, A. SoundNet: Learning Sound Representations from Unlabeled Video. In Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS'16), Barcelona, Spain, 5–10 December 2016; Curran Associates Inc.: Red Hook, NY, USA, 2016; pp. 892–900.
22. Giannakopoulos, T.; Perantonis, S. Recognition of Urban Sound Events using Deep Context-Aware Feature Extractors and Handcrafted Features. In Proceedings of the IFIP International Conference on Artificial Intelligence Applications and Innovations, Rhodes, Greece, 25–27 May 2018.
23. Martin-Morato, I.; Mesaros, A.; Heittola, T.; Virtanen, T.; Cobos, M.; J. Ferri, F. Sound Event Envelope Estimation in Polyphonic Mixtures. In Proceedings of the 2019 IEEE International Conference on Acoustics (ICASSP 2019), Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 935–939. [CrossRef]
24. Grais, E.; Umut Sen, M.; Erdogan, H. Deep neural networks for single channel source separation. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2013; doi:10.1109/ICASSP.2014.6854299. [CrossRef]
25. Vera-Diaz, J.; Pizarro, D.; Macias-Guarasa, J. Towards End-to-End Acoustic Localization Using Deep Learning: From Audio Signals to Source Position Coordinates. *Sensors* **2018**, *18*, 3418. [CrossRef] [PubMed]