Breaking Batch Normalization for better explainability of Deep Neural Networks through Layer-wise Relevance Propagation

Mathilde Guillemot[1,2] Catherine Heusele[2] Rodolphe Korichi [2] Sylvianne Schnebert [2] Liming Chen [1]

[1] LIRIS - UMR 5205 CNRS
[2] LVMH Recherche - Parfums et cosmetiques

mathilde.guillemot@ec-lyon.fr

Abstract

The lack of transparency of neural networks is a significant disadvantage, limiting their interpretability and their practical applications. The Layer-wise Relevance Propagation is a technique building heat-maps representing the relevance of each input in the model's decision. The relevance spreads backward from the last to the first layer of the Deep Neural Network. With those kinds of methods, deep neural networks are no longer black boxes and the contribution of each input element can be determined. However, Layer-wise Relevance Propagation has mostly be tested on network without batch normalization layers, in this work we suggest a method that adapts Layer-wise Relevance Propagation technique to normalization layers. Specifically, we build an equivalent network fusing normalization layers and convolutional or fully connected layers. Heat-maps obtained with our method on MNIST, CIFAR-10 and ImageNet data-sets are more accurate than the state of the art for convolutional layers. Our study also advises against using Layer-wise Relevance Propagation with networks including a combination of connected layers and normalization layers.

Keywords

Layer-wise relevance propagation, Batch normalization, Convolutional neural networks, Fully-connected neural network.

1 Introduction

Neural networks are nonlinear models showing high performances on various problems like computer vision [15, 8, 16], speech synthesis [38], material science and quantum physics [22, 4, 29] or even cancer research [2, 13]. Nevertheless, they are not easily interpretable and until recently DNN were considered as black boxes [19]. The inner working of these models is hard to understand, indeed DNN are complex models formed with multiple connections between neurons combined with nonlinear operations. Thus, human expert cannot perform critical analysis and the results provided by those models as good as they can be, must be trusted blindly. Explainable Artificial Intelligence (XAI) is developed to overcome this issue.

Multiple interests of interpretability can be found [28]. Firstly, for delicate domains as autonomous car [3] or medical diagnosis [2] consequences of a mistake can be disastrous. In such cases, before delivering any result, one must ensure that the classifier works as expected and takes decisions based on relevant information [36]. Then, interpretability also enables to improve a classifier by addition of human experience *i.e.*, to analyze the learning algorithm's errors. And the reverse operation, *i.e.*, learning from the model, opens interesting perspectives. For instance, the AlphaGo algorithm [33] played during its game moves that humans are not used to. Finally, when neural networks achieve good results for a task a human cannot perform, specifically in physics or chemical areas, interpretability would therefore enable to discover new principles as finding new genes linked to cancer or identify binding sites [30].

Several methods have been developed in order to deal with the explainability of DNN. First introduced methods build saliency maps [34] or visualizations of patches that maximally activate neurons [6]. Other suggested gradient methods to explain reasons why images were misclassified [31]. LIME [24] or SHAP [20] justify the predictions though an explainable classifier locally around the prediction. DeepLIFT [32] decomposes the prediction by assigning the differences of contribution scores between the activation of each neuron to its reference activation. Layer-wise relevance propagation (LRP) is based on the deep Taylor decomposition [21], decomposing the activation of a neuron as the contributions from its outputs. LRP has shown interesting results, allowing interpretation on various neural networks [26, 21, 11].

Contributions. In this work, we propose an improvement of the Layer-Wise Propagation (LRP) method [1]. LRP is a post-hoc interpretability method, implemented after the model training [24]. It explains the model decisions one sample at a time. LRP propagates backward the relevance though all layers from output results to input features. The propagation follows different rules for convolutional

layers, pooling layers, etc. However, it is not clear how to apply it to normalization layers [23], and in some recent work, normalization layers are bypassed during the relevance back-propagation [21]. We develop a method to easily include normalization layers to LRP method. We prove that properly fusing batch normalization (BN) [12] with another layer enables to integrate BN rather than bypassing it. We also show the extension of this method to other normalization layers. To assess the improvement brought by our method, it is tested on three data-sets MNIST [17], CIFAR-10 [14] and ImageNet [5]. Different networks architecture including fully connected neural networks[25] and convolutional neural networks [18] are tested. We demonstrate that relevance computation with BN obtains better results than ignoring BN for convolutional neural networks. Relatively to fully connected network, we reserve our conclusion since LRP seems not to be compatible with fully connected networks combined with batch normalization.

Related work. There are few works on the integration of batch normalization for method decomposing and creating explanation based on the linearization of the respective compounds. One paper has already taken on the subject of batch normalization layers with the LRP technique [11]. In this paper, the performance of three deep neural network classifiers (MobileNet, ResNet and DenseNet) are analyzed testing different decomposition rules adapted for LRP. From this analysis, a new batch normalization rule is proposed to provide more robust performances across various deep network architectures. More generally, some approaches have been proposed but not tested as absorbing BN and the adjacent layer without changing the function, or considering BN as one or two linear layers [27].

2 Background and Notation

We consider supervised learning tasks. Since LRP only processes samples one by one, notation of input features ℓ is simply referred as $x^{(\ell)}$. The weights and biases between neuron i belonging to layer l, $x_i^{(\ell)}$ and neuron j in to layer (ℓ+1), $x_j^{(\ell+1)}$ are respectively written down $w_{ij}^{(\ell,\ell+1)}$ and $b_j^{(\ell,\ell+1)}$. Also, ReLU function *i.e.*, $max(.,0)$ is written $(.)^+$.

2.1 Fully connected layer

Fully connected layers connect every neuron of one layer to every neuron of the next layer. Equation 1 gives the expression of the output neurons as a function of the input neurons and the fully connected layer parameters.

$$\forall \ell, \forall i, \forall j, x_j^{(\ell+1)} = \sum_i w_{ij}^{(\ell,\ell+1)} x_i^{(\ell)} + b_j^{\ell} \qquad (1)$$

2.2 Convolutional Layer

Unlike fully connected layers, convolutional layers contain a set of filters. Each filter is convolved with the input layer (ℓ) to compute an activation map. The filter is slided across the width and height of the input and the dot products between the input and the filter are computed at every spatial position (2).

$$x^{(\ell+1)} = w^{(\ell,\ell+1)} * x^{(\ell)} + b^{(\ell,\ell+1)} \qquad (2)$$

2.3 Batch Normalization

Batch normalization [12], is a trick commonly used to improve the training of deep neural networks, accelerating learning phase and showing better accuracy. Its success leads various deep learning structure to incorporate batch normalization [7] [10]. During the learning phase, batch normalization avoids problems related to back-propagation. It prevents the gradient from explosion and the vanishing gradient problem by keeping data in bounded intervals. During the test phase, batch normalization is performed using constant variance and constant mean. Equation 3 expresses the output ${x_i^{(\ell)}}_{norm}$ of a BN layer (ℓ) as a function of the input $x_i^{(\ell)}$ during the test phase.

$${x_i^{(\ell)}}_{norm} = \gamma^{(\ell)} \frac{x_i^{(\ell)} - \mu_{run}^{(\ell)}}{\sigma_{run}^{(\ell)}} + \beta^{(\ell)} \qquad (3)$$

with $\gamma^{(\ell)}$ and $\beta^{(\ell)}$ respectively weights and biases of the layer.

2.4 Layer-wise Relevance Propagation

Method. LRP is applied once the network is learned. It consists in finding the relevance R_i of each input feature $x_i^{(1)}$, propagating backward the relevance information from the output until the input. The relevance obeys to conservation rule from one layer to another (4).

$$\forall \ell, \sum_i R_i^{(\ell)} = \sum_j R_j^{(\ell+1)} \qquad (4)$$

Equation 5 translates $R_i^{(\ell)}$, the relevance of neuron i in layer ℓ, as the sum of all the contributions of neurons communicating with it.

$$\forall \ell, \forall i, R_i^{(\ell)} = \sum_j R_{i \leftarrow j}^{(\ell+1)} \qquad (5)$$

Several rules are established satisfying equations 4 and 5 to propagate the relevance from a layer to the previous one [23]. In this paper we will use two of them depending on the input domain.

— Rule 1 : If the neuron value $x_i^{(\ell)}$ is positive. The relevance of the neuron i of the layer ℓ is computed as in equation 6

$${R_i}^{(\ell)} = \sum_j \frac{{x_i}^{(\ell)} {w_{ij}}^{(\ell,\ell+1)^+}}{\sum_k {x_k}^{(\ell)} {w_{kj}}^{(\ell,\ell+1)^+}} {R_j}^{(\ell+1)} \qquad (6)$$

with ${w_{ij}^{(\ell,\ell+1)}}^+ = max(0, w_{ij}^{(\ell,\ell+1)})$

— Rule 2 : If the neuron values $x_i^{(\ell)}$ range between ℓ_i and h_i

$$R_i^{(\ell)} = \sum_j k_{ij\ell} R_j^{(\ell+1)}$$

$$k_{ij\ell} = \frac{x_i^{(\ell)} w_{ij}^{(\ell,\ell+1)} - \ell_i^{(\ell)} w_{ij}^{(\ell,\ell+1)+} - h_i^{(\ell)} w_{ij}^{(\ell,\ell+1)-}}{\sum_k x_k^{(\ell)} w_{kj}^{(\ell,\ell+1)} - \ell_k^{(\ell)} w_{kj}^{(\ell,\ell+1)+} - h_k^{(\ell)} w_{kj}^{(\ell,\ell+1)-}} \tag{7}$$

with $w_{ij}^{(\ell,\ell+1)+} = max(0, w_{ij}^{(\ell,\ell+1)})$ and $w_{ij}^{(\ell,\ell+1)-} = min(0, w_{ij}^{(\ell,\ell+1)})$

Relevance R computed with LRP is pictured as a heat-map *i.e.*, a visualization technique highlighting pixels which support the classification decisions [34] [1].

Practical Considerations. The input of LRP corresponds to the raw output of the network *i.e.*, before SoftMax activation function. The last activation function is a SoftMax the others chosen nonlinear activation functions are ReLU functions. Therefore, for all $l \neq 0$ and for all i, $x_i^{(\ell)} > 0$ and the relevance is propagated according to equation 6. As we work with images, pixel inputs $x_i^{(0)}$ are bounded between 0 and 255 (or -1 and 1 if a scale operation is applied) and rule 2 (7) is applied.

Pooling layers are easily handled, being considered as reLU detection layers.

All biases are recommended to be either zero or negative [23]. When the condition is not filled, biases are considered as neurons and their contribution is added on the denominator of equation 6 or 7 [21].

All results shown are part of the test set.

3 Batch normalization in the LRP relevance computation

Various pre-trained networks [37] [9] [10] include normalization layers and show good results on various tasks. Such networks support LRP with approximation, but we expect better results with a suitable way to handle BN layers with LRP. We propose a new method to obtain the relevance heat-maps of a DNN classifier with BN layers.

The normalization layer is applied, before or after the activation. The general idea of our method is to fuse the batch normalization layer with the closest convolutional or fully connected layer (see Figure 1) into a single convolutional or fully connected layer simply by modifying their parameters. $w_{ij}^{(\ell,\ell+1)'}$ and $b_{ij}^{(\ell,\ell+1)'}$ are the parameters of this new layer.

3.1 Fully connected neural network

Combining the two equations 1 and 3, we show that the combination of a BN layer and a FC layer is equivalent to a single fully connected layer by adapting its weight and bias parameters.

BN after activation. (see 1, graph a))

If a BN fuses with a FC layer, equation 8 describes the new propagation rule.

$$\forall \ell, \forall j, x_j^{(\ell+1)} = \sum_i w_{ij}^{(\ell,\ell+1)} \left(\gamma_i^{(\ell)} \frac{x_i^{(ell)} - {\mu_i^{(\ell)}}_{run}}{{\sigma_i^{(\ell)}}_{run}} + \beta^{(\ell)} \right) + b_j^{(\ell,\ell+1)} \tag{8}$$

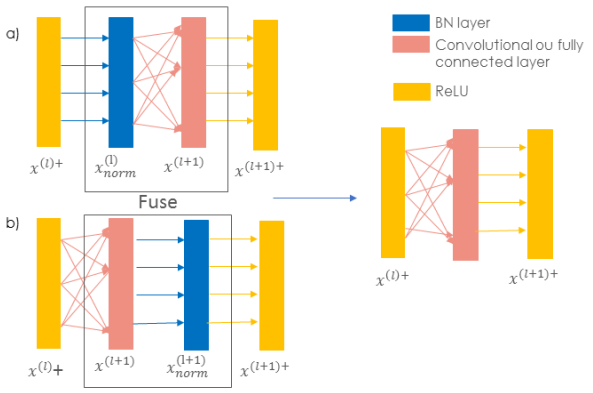

FIGURE 1 – Fusion of the normalization layer with another layer in the two different configurations a) BN after activation b) BN before activation

$$\forall \ell, \forall j, x_j^{(\ell+1)} = \sum_i \left[\frac{\gamma_i^{(\ell)} w_{ij}^{(\ell,\ell+1)}}{{\sigma_i^{(\ell)}}_{run}} \right] x_i^{(\ell)} + \left[b_j^{(\ell,\ell+1)} + \sum_i w_{ij}^{(\ell,\ell+1)} \left(\beta^{(\ell)} - \frac{\gamma_i^{(\ell)} {\mu_i^{(\ell)}}_{run}}{{\sigma_i^{(\ell)}}_{run}} \right) \right] \tag{9}$$

By identification, we find weight and bias terms of the fuse layer respectively in the first and second bracket of equation 9.

BN before activation. (see 1, graph b))

Similarly, we find out that a FC layer and a BN layer can be fused into a single FC layer with the following parameters.

$$w_{ij}^{(\ell,\ell+1)'} = \frac{\gamma_j^{(\ell+1)} w_{ij}^{(\ell,\ell+1)}}{{\sigma_j^{(\ell+1)}}_{run}} \tag{10}$$

$$b_j^{(\ell,\ell+1)'} = \beta_j^{(\ell+1)} + \gamma_j^{(\ell+1)} \frac{b_j^{(\ell,\ell+1)} - {\mu_j^{(\ell+1)}}_{run}}{{\sigma_j^{(\ell+1)}}_{run}} \tag{11}$$

3.2 Convolutional neural network

When the batch normalization is combined with a fully connected layer, all input neurons receive a different normalization *i.e.*, $\gamma^{(\ell)}$, $\beta^{(\ell)}$, $\sigma_{run}^{(\ell+1)}$ and $\mu_{run}^{(\ell)}$ are vectors which size is equal to the neurons vector's one. With a convolutional layer, the same normalization is applied to all neurons of a channel (see Figure. 2). BN parameters are simplified :

$$\begin{aligned} \forall i, \gamma_i^{(\ell)} &= \gamma^{(\ell)} \\ \beta_i^{(\ell)} &= \beta^{(\ell)} \\ {\mu_i^{(\ell)}}_{run} &= {mu^{(\ell)}}_{run} \\ {\sigma_i^{(\ell)}}_{run} &= {\sigma^{(\ell)}}_{run} \end{aligned} \tag{12}$$

Consequently, we rewrite equation 3 removing unnecessary terms.

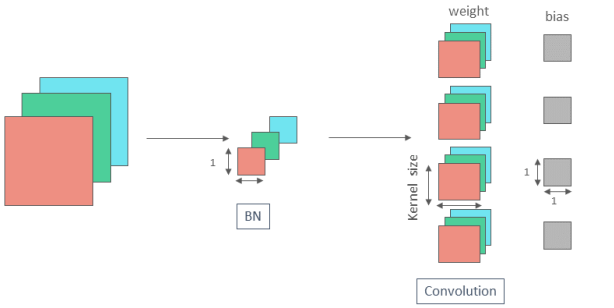

FIGURE 2 – Scheme of a batch normalization applied before a convolutional layer

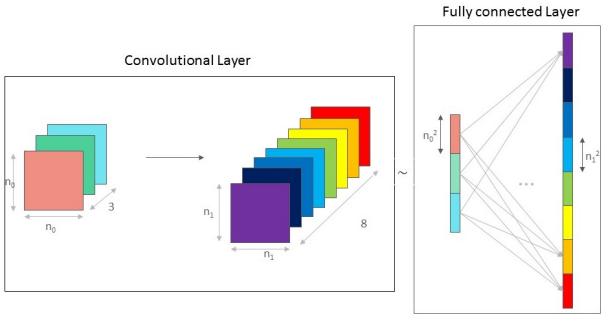

FIGURE 3 – Example of a convolution layer and its equivalent as a fully connected layer after a flatten operation

BN after activation. Using equation 2 and 3 and proceeding similarly to the previous sections, a convolutional layer combined with a BN layer can be reduced to a single convolution layer which weights and biases are expressed as :

$$w^{(\ell,\ell+1)'} = \frac{w^{(\ell,\ell+1)}\gamma^{(\ell)}}{\sigma^{(\ell)}{}_{run}} \tag{13}$$

$$b^{(\ell,\ell+1)'} = b^{(\ell,\ell+1)} + w^{(\ell,\ell+1)}(\beta^{(\ell)} - \frac{\gamma^{(\ell)}\mu^{(\ell)}{}_{run}}{\sigma^{(\ell)}} \tag{14}$$

BN before activation. Similarly, if we apply BN between the convolution and the non-linearity, the weights and biases of the resulting fused convolutional layer are expressed as :

$$w^{(\ell,\ell+1)'} = \frac{w^{(\ell,\ell+1)}\gamma^{(\ell)}}{\sigma^{(\ell)}_{run}} \tag{15}$$

$$b^{(\ell,\ell+1)'} = \beta^{(\ell+1)} + \gamma^{(\ell+1)}\frac{b^{(\ell,\ell+1)} - \mu^{(\ell+1)}_{run}}{\sigma^{(\ell+1)}} \tag{16}$$

4 More complex normalizations for convolution layers - from convolutional to fully connected layer

Batch normalization at test time consists in applying the same mean and variance for all coefficients of an input (see Figure 2). For other normalizations, parameters (mean, variance etc.) may not be constant. The simplifications expressed in equation 12 are no longer practical. Another way to proceed must be found.

We reduce the convolutional layer to a fully connected layer with weight W_{fc} and bias B_{fc} and then apply the known results on FC. In practice two dimensional inputs and outputs of the layer are flatten (see Figure 3). And a weight matrix of the new created fully connected layer is filled with the coefficients of the different kernels.

For each connection between an input flatten channel and an output flatten channel, a weight matrix is created. Given c_0, c_1 respectively the number of input and output channels, for all i $\in [1, c_0]$ and for all j $\in [1, c_1]$, a weight matrix w_{ij} and a bias vector b_j are filled. The final weight matrix and bias vector are the concatenation of those sub-matrix :

$$W_{fc} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,c_0} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,c_0} \\ \vdots & \vdots & \ddots & \vdots \\ w_{c_1,1} & w_{c_1,2} & \cdots & w_{c_1,c_0} \end{pmatrix} \text{ and } B_{fc} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{c_1,} \end{pmatrix}$$

The matrix w_{ij} and vector b_j coefficients are expressed thanks to kernel coefficients of the concerned couple of input channel i and output channel j. To illustrate, we consider a convolutional layer with an input channel with 5*5 images and a 2*2 kernel, no padding and no stride. The coefficients of the kernel are written α_i and the pixels are noted p_i. The convolution can be drawn as :

$$W_{fc} = \begin{pmatrix} p_1 & p_2 & \cdots & p_5 \\ p_6 & p_7 & \cdots & p_{10} \\ \vdots & \vdots & \ddots & \vdots \\ p_{21} & p_{22} & \cdots & p_{25} \end{pmatrix} * \begin{pmatrix} \alpha_1 & \alpha_2 \\ \alpha_3 & \alpha_4 \end{pmatrix}$$

In this case, the sub-matrix weight matrix w_{ij} is

$$w_{ij} = \begin{pmatrix} \alpha_1 & \alpha_2 & 0 & 0 & 0 & \alpha_3 & \alpha_4 & 0 & \cdots & 0 \\ 0 & \alpha_1 & \alpha_2 & 0 & 0 & 0 & \alpha_3 & \alpha_4 & \cdots & 0 \\ 0 & 0 & \alpha_1 & \alpha_2 & 0 & 0 & 0 & \alpha_3 & \cdots & 0 \\ 0 & 0 & 0 & \alpha_1 & \alpha_2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_1 & \alpha_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \alpha_4 \end{pmatrix}$$

Performing those operations, we can convert any convolution layer into a fully connected layer and then apply the different results found on FC layers.

After briefly introducing the LRP method, we have seen how it can theoretically be adapted to batch normalization. In the next section, several experiments are conducted to analyze the impact of taking the batch normalization into account on both fully connected and convolutional layer. The previous section has also shown how other normalizations can be used when working with LRP, this last part is not studied experimentally.

5 Experiments

In this section, we show the results obtained by applying our method to include batch normalization into the LRP method. Three data-sets are tested with our method. MNIST is an interesting data-set for this work, because it is simple enough to achieve good results with fully connected layer but as soon as entrance data are two dimensional images, it can also be treated with a convolutional network. However, MNIST data-set is particularly simple, the method is also tested on other data-sets (CIFAR-10 and ImageNet) to consolidate our results. Because CIFAR-10 are more complex data, a network composed with fully connected layers will not give usable results. Consequently, only convolutional layers are studied. However, for our task, MNIST and CIFAR-10 are trivial data-sets since the relevance largely corresponds to the foreground object in the image leading us to a more extensive study on the ImageNet data-set.

Images from the two first data-sets are normalized such as all pixel values are contained in [-1,1] (see equation 17).

$$\forall i, \forall j, x_{ij} \in [0, 255], x_{ij_{norm}} = \frac{\frac{x_{ij}}{255} - 0.5}{0.5} \qquad (17)$$

The classical normalization for ImageNet is performed, the RGB values of images are normalized using [0.485, 0.456, 0.406] as mean and [0.229, 0.224, 0.225] as standard deviation.

With toy data-sets, the heat-maps are expected to display the same pixels as a human eye would do. A satisfying explainable method creates a heat-map in which the contours and important shapes of objects are intensely red while background elements and insignificant detailed are white.

The identity rule for batch- normalization layers (*i.e.*, bypass normalization layers) [23], it is the baseline we choose to measure our contribution.

5.1 MNIST

Fully connected neural network. Three fully connected neural networks only composed with BN and FC layers are studied. Fc1 is only composed with FC layers, Fc2's architecture is similar to Fc1 adding BN layers after activation function. Fc3 presents a little different architecture and uses a BN layer after FC operation and before non-linearity. Figure 4 a) and b) detail those networks.

The networks' performances are measured with the accuracy criterion. All networks give good results, with an accuracy between 97 and 99.24% (see Table 1).

Convolutional neural network. The procedure applied for the fully connected network is repeated with a convolutional network architecture. Two convolutional networks are built the Conv1 with four convolutional layers, and the Conv2 adding a batch normalization layer before every convolutional layer. Those networks architectures are detailed on Figure 4 c).

Some of the heat-maps obtained with LRP method are shown in 5. Fc2 and Conv2 give two different heat-maps,

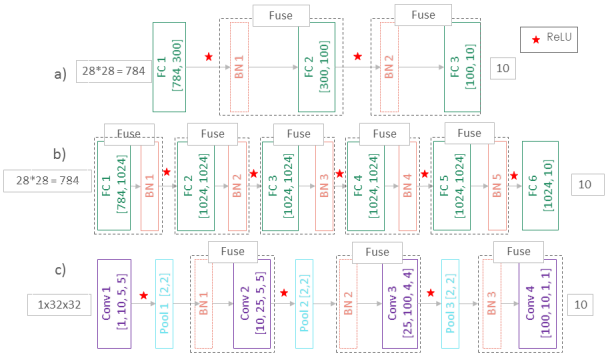

FIGURE 4 – Different networks configurations tested on MNIST data-set. The **a)** figure is the network Fc2 with BN layers before FC layers, Fc1 has the same architecture as Fc2 without BN layers **b)** is the fully connected layer Fc3 with FC before BN layers. Finally network **c)** is the architecture of the convolutional network Conv2, Conv1 corresponds to the architecture of conv2 without the BN layers

Name	Type	BN in NN	Position of BN	Accuracy
Fc1	FC	No		0.9781
Fc2	FC	Yes	Before FC	0.9742
Fc3	FC	Yes	After FC	0.9831
Conv1	Conv	No		0.9903
Conv2	Conv	Yes	Afer conv	0.9924

TABLE 1 – Accuracy computed on MNIST data-set

the first (Figure 5 column 'Fc2 w/ BN' or 'Conv2 w/ BN') using the method developed in this paper, the other (Figure 5 column 'Fc2 w/o BN' or 'Conv2 w/o BN') bypassing the normalization layers *i.e.*, the baseline.

5.2 CIFAR-10

The chosen network architecture for the study of CIFAR-10 is composed with seven convolutional layers, each of them is directly followed by a batch normalization layer. It ends with a fully connected layer as it is usually done in classification problems involving convolutional neural networks. Four pooling layers are added to down sampling the intermediate results. More complete information on kernel sizes, and pooling layers location is available at Figure 6. The network reaches an accuracy of 0.9378 on the test set. It leads to two different heat-maps to compare : the first one gives the result with our method meaning considering BN layers while the second one bypasses the BN layers.

Some result examples are shown in Figure 7.

5.3 ImageNet

Concerning the last studied data-set, we use a pre-trained convolutional neural network VGG-16 [35], the detailed architecture is the D version of the network described in the article. In this configuration, the batch normalization layer is placed between the convolutional layer and the nonlinear activation function. For this model the performances achie-

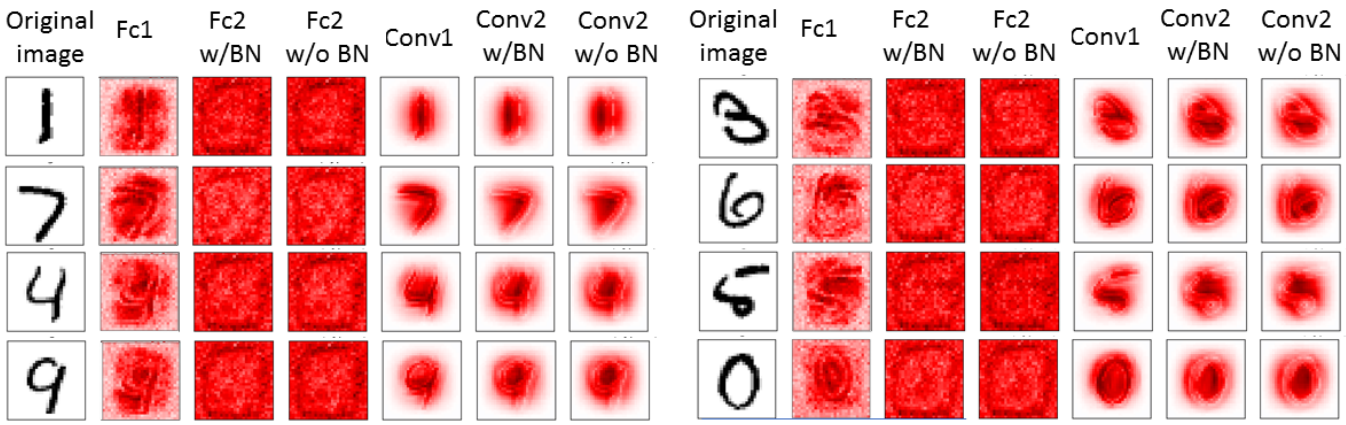

FIGURE 5 – Heat-maps obtained by applying LRP to MNIST data-set through different networks. The first column **Original image** is the raw image, **Fc1** is the result obtained with the first FC network Fc1, **Fc2 w/BN**, is the heat-map obtained with the Fc2 network architecture taking BN layers into account during the LRP phase, on the contrary **Fc2 w/o BN** column is the heat-map obtained with Fc2 bypassing normalization layers while relevance backpropagation. Similarly, the heat-map resulting from the Conv1 architecture (without BN layers) is given in column **Conv1**, and columns **Conv2 w/o BN** and **Conv2 w/o BN** show respectively the Conv2 network heat-maps result with BN and bypassing BN layers for LRP.

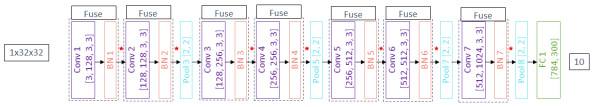

FIGURE 6 – Configuration architecture of the network trained on CIFAR-10 data-set

ved on the test set are 26.63% top-1 error and 8.50% top-5 error. The study configuration is close to the CIFAR-10 one's, consequently we also generate two heat-maps, the baseline comparison using the identity rule for the batch normalization layers and the classical LRP rules after fusing the batch normalization layer with the closest convolutional layer. Figure 8 displays some results on ImageNet data-set.

This previous section explains experiments run on MNIST, CIFAR-10 and ImageNet data-sets with either fully connected or convolutional layers. In the next section, we discuss the results of these experiments, particularly through the interpretation of Figures 5, 7 and 8.

5.4 Results

The analysis presented here is based on qualitative analysis *i.e.*, we compare visually the red intensity difference between pixels of interest and pixels belonging to the background.

Better results obtained with convolutional layers than fully connected layers. Globally, convolutional layers (see Figure 5, columns "Conv1", "Covn2 w/BN" and "Conv2 w/o BN") give better results than fully connected (see Figure 5, columns "Fc1", "Fc2 w/BN" and "Fc2 w/o BN") layers. Relevance computed with convolutional networks marks more the difference between background and figures. This is a logical conclusion since convolutional networks give better results and there is a strong connection between LRP heat-map quality and network efficiency. [11].

Using BN with fully connected layer provides poor results for the relevance. Concerning the fully connected network, the network learned without batch normalization layer captures a good relevance information (see Figure 5, column Fc1).

The results on the network learned with batch normalization layers are bad considering batch normalization or not during the relevance phase (see Figure 5 columns "Fc2 w/BN" and "Fc2 w/o BN"). In this case heat-maps highlight all pixels of the image center. The batch normalization interferes in the relevance computation and takes precedence over the figure relevance signal.

For the fully connected network Fc3 where unlike Fc2, batch normalization is placed after FC layers, the explicit results are not given here but are very similar to Fc2 results. With BN, whatever the configuration chosen *i.e.*, placed before or after activation in the architecture and bypassed during relevance propagation or using our method, results are unusable and no relevant. This might be explained by MNIST data, information is always in the center of the image. LRP method should be used carefully when dealing with fully connected layers combined with batch normalization layers.

Relevance obtained with a convolutional network built without BN highlights all pixel of the object while convolutional network with BN highlights contours. Heat-maps computed for Fc2 bypassing BN or not during relevance propagation are similar, in this section we will focus only on the columns "Conv1" and "Conv2 w/BN" of Figure 5.

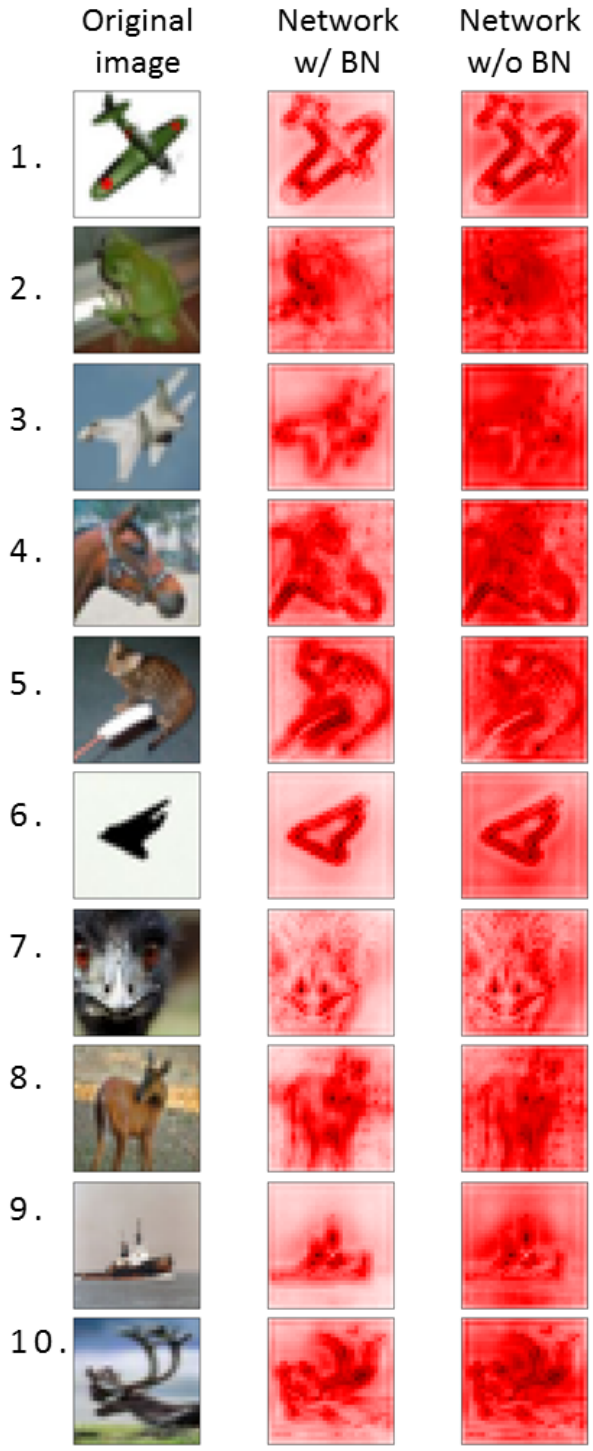

FIGURE 7 – Heat-maps showing the entrance pixel relevance in the decision of a convolutional network using the LRP method. The first column **Original Image** is the image form CIFRA-10 data-set studied, the **Network w/ BN** are results of relevance propagation with the whole model including BN layers, in comparison column **Network w/o BN** shows the results for the same image still using LRP but bypassing BN layers.

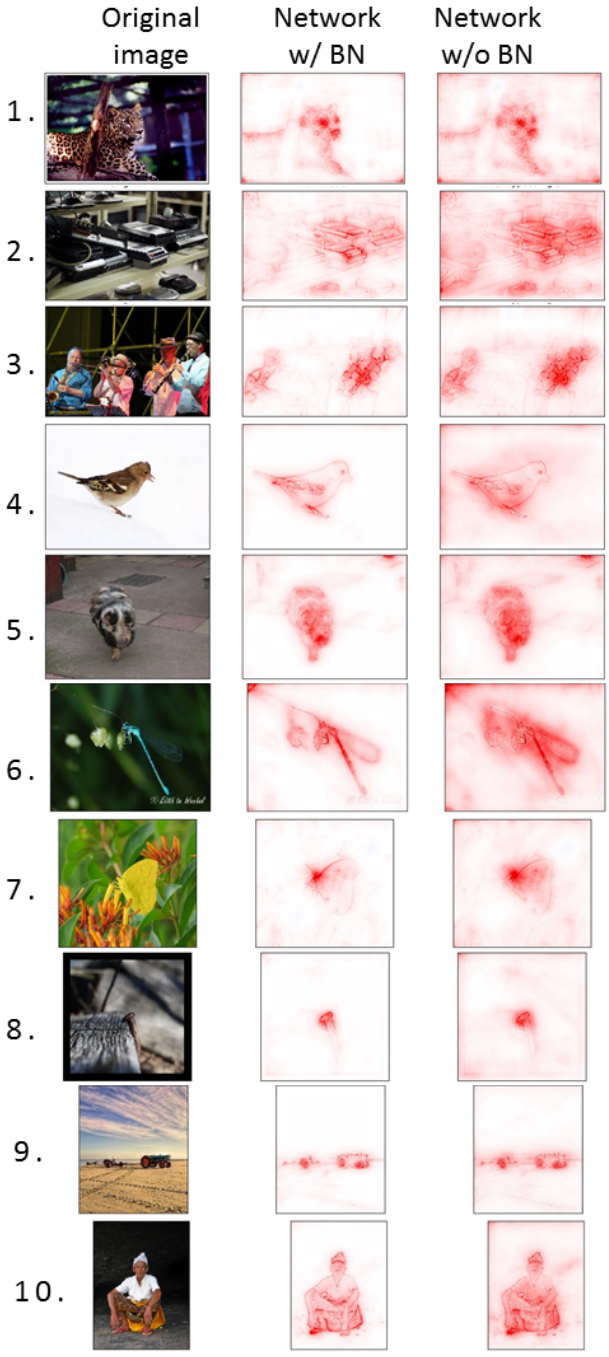

FIGURE 8 – Heat-maps showing the entrance pixel relevance in the decision of a convolutional network using the LRP method. The first column **Original Image** is the image form ImageNet data-set studied, the **Network w/ BN** are results of relevance propagation with the whole model including BN layers, in comparison column **Network w/o BN** shows the results for the same image still using LRP but bypassing BN layers.

About convolutional networks, results are very satisfying, and pixels of interest are well localized. However there are differences between the two heat-maps obtained by applying LRP with a model without BN and with a model with BN. Looking with attention at columns Conv1 and Conv2 w/ BN of the Figure 5, it appears that the relevance computed from Conv1 gives importance to the pixels composing the figure while using a model with BN, the edges are spotted and the internal pixels are completely white.

Relevance heat-maps computed with our method gives more accurate results than the baseline. When BN is employed jointly with convolutional network, relevant pixels found with LRP are the edge of the object's shape.

Concerning MNIST data, there are no big differences between the two last columns of Figure 5. But little nuances are observable specifically for figure comporting a loop like 0 or 9, inside the loop, the red color is eased by taking the BN into account during the LRP computation.

For CIFAR-10 and ImageNet data-sets (see Figures 7 and 8), on each example, results obtained when BN is not bypassed are significantly better *i.e.*, the contrast between the background and the object is more apparent when BN is introduced in relevance calculation.

When the **image is simple** (*i.e.*, only one object on the picture) **and the background is uniform** : with the explanation done on the ImageNet data-set, heat-map obtained with our method are nearly perfect (Figure 8 images 4. and 10.) whereas using the identity rule to treat batch normalization layers leads to noisy heat-maps, with many background pixels qualified as pixel of interest. Concerning the CIFAR-10 data-set, different cases can be distinguished. When the background is uniform and has a very different color than the object, there is an improvement using BN in relevance, but the result bypassing BN is already good, this can be observed on the first and sixth images of Figure 7). When the background is uniform, but its color is close to the object's color, LRP using BN gives equivalent results to the ones obtained when the background color was more distant. We can observe this on the third image, the fifth, seventh, and ninth ones. On the contrary, in this case where the difference between background and object is not that clear, not using BN in the relevance leads to medium result. The shape of the object is distinguishable from the rest of the image, but the contrast between the intensity of pixel belonging or not to the object is not pronounced.

When the **image is simple** (*i.e.*, only one object on the picture) **but the background is not uniform** : for the CIFAR-10 data-set, LRP without BN gives poor results as for the frog (second image), the horse (fourth image) or deer (last image) in Figure 7. The results for LRP taking BN into account is not as good as the previous ones but are much better than relevance when BN is ignored. On the contrary, for the ImageNet data-set, heat-maps obtained bypassing the batch normalization layer give a good explanation but using our method improves the heat-maps and most of the background pixels are turned off. We can observe this on the Figure 8, images 1., 5., 6. and 9.

Finally, we study **complex scenes** from the ImageNet data-set where the class of interest is melted with other objects (*e.g.*, Figure 8, images 2., 3. and 8.). In those cases, with our method pixels of interest almost only belong to objects of interest, the tape players on image 2., the saxes on image 3., the butterfly on image 7. and the nail for the image 8. Similarly, to our previous observations, results given bypassing the BN layer are not bad and turn on the pixels belonging to the target classes, but the other elements in images are also highlighted. This is particularly sensitive on the second image of Figure 8.

6 Conclusion

In this work we propose a method to properly build heat-maps with LRP on network containing normalization. From the combination of a fully connected layer or a convolutional layer and a normalization layer we create a new layer on which we can easily apply LRP. We explicit parameters of this new layer for BN used before or after activation. In practice, the method is tested on two trivial data-sets : MNIST and CIFAR-10 and more complex scenes from ImageNet data-set. Several conclusions emerge from this study, mainly we show an improvement using our method compared with baseline *i.e.*, bypassing normalization layers. Our study seems to show that the more inputs will be complex, the more benefits achieved with our method will be important. Furthermore, we have noticed that using LRP with a fully connected layer containing BN leads to irrelevant heat-maps in the case of MNIST data-set. There is no proof that this observation is true for all data-sets, care must be taken with this configuration. For future work, other normalization can be tested to evaluate the impact of our method and the improvement provided. The case of fully connected layer should be examined in detail to ensure that this very particular data-set is not involved in the bad results. Finally, in this work, we chose to evaluate the heat-maps qualitatively, the development of a method to measure the accuracy of a heat-map would give a more reliable comparison between all our results and might also be a research track.

Références

[1] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7) :e0130140, July 2015.

[2] A. Binder, M. Bockmayr, M. Hägele, S. Wienert, D. Heim, K. Hellweg, A. Stenzinger, L. Parlow, J. Budczies, B. Goeppert, D. Treue, M. Kotani, M. Ishii, M. Dietel, A. Hocke, C. Denkert, K. Müller, and F. Klauschen. Towards computational fluorescence microscopy : Machine learning-based integrated prediction of morphological and molecular tumor profiles. *CoRR*, abs/1805.11178, 2018.

[3] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, and U. Muller. Explaining how a deep neural network trained with end-to-end learning steers a car. *CoRR*, abs/1704.07911, 2017.

[4] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*, 3(5), 2017.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet : A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2009.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv e-prints*, page arXiv :1311.2524, Nov 2013.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2016.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets : Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[10] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[11] L. Y. W. Hui and A. Binder. BatchNorm decomposition for deep neural network interpretation. In *Advances in Computational Intelligence*, pages 280–291. Springer International Publishing, 2019.

[12] S. Ioffe and C. Szegedy. Batch normalization : Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[13] B. Korbar, A. M. Olofson, A. P. Miraflor, C. M. Nicka, M. A. Suriawinata, L. Torresani, A. A. Suriawinata, and S. Hassanpour. Looking under the hood : Deep neural network visualization to interpret whole-slide image analysis outcomes for colorectal polyps. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, July 2017.

[14] A. Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[16] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553) :436–444, May 2015.

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324, Nov 1998.

[18] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio. *Object Recognition with Gradient-Based Learning*, pages 319–345. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[19] Z. C. Lipton. The mythos of model interpretability. *ArXiv*, abs/1606.03490, 2016.

[20] S. Lundberg and S. Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.

[21] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65 :211–222, May 2017.

[22] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *New Journal of Physics*, 15(9) :095003, sep 2013.

[23] G. Montavon, W. Samek, and K. Müller. Methods for interpreting and understanding deep neural networks. *CoRR*, abs/1706.07979, 2017.

[24] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you ?" : Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088) :533–536, Oct. 1986.

[26] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Muller. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11) :2660–2673, Nov. 2017.

[27] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, editors. *Explainable AI : Interpreting, Explaining and Visualizing Deep Learning*. Springer International Publishing, 2019.

[28] W. Samek, T. Wiegand, and K. Müller. Explainable artificial intelligence : Understanding, visualizing and interpreting deep learning models. *CoRR*, abs/1708.08296, 2017.

[29] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8(1), Jan. 2017.

[30] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko. Quantum-chemical insights

from deep tensor neural networks. *Nature Communications*, 8 :13890, Jan 2017.

[31] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-cam : Why did you say that ? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.

[32] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. *CoRR*, abs/1704.02685, 2017.

[33] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529 :484–489, 01 2016.

[34] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep Inside Convolutional Networks : Visualising Image Classification Models and Saliency Maps. *arXiv e-prints*, page arXiv :1312.6034, Dec 2013.

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[36] P. Stock and M. Cisse. ConvNets and ImageNet Beyond Accuracy : Understanding Mistakes and Uncovering Biases. *arXiv e-prints*, page arXiv :1711.11443, Nov 2017.

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[38] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet : A generative model for raw audio, 2016.