Enhancing LSTM for sequential image classification by modifying data aggregation

Zdenko Takáč
Inst. of Information Engineering, Automation and Mathematics
Slovak University of Technology in Bratislava
Bratislava, Slovakia
zdenko.takac@stuba.sk

Mikel Ferrero-Jaurrieta
Dept. of Statistics, Computer Science and Mathematics
Public University of Navarre
Pamplona, Spain
mikel.ferrero@unavarra.es

Ľubomira Horanská
Inst. of Informat. Engineering, Automation and Mathematics
Slovak University of Technology in Bratislava
Bratislava, Slovakia
lubomira.horanska@stuba.sk

Naďa Krivoňáková
Inst. of Informat. Engineering, Automation and Mathematics
Slovak University of Technology in Bratislava
Bratislava, Slovakia
nada.krivonakova@stuba.sk

Graçaliz Pereira Dimuro
Dept. of Statistics, Computer Science and Mathematics
Public University of Navarre
Pamplona, Spain
gracaliz.pereira@unavarra.es

Humberto Bustince
Dept. of Statistics, Computer Science and Mathematics
Public University of Navarre
Pamplona, Spain
bustince@unavarra.es

***Abstract*—Recurrent Neural Networks (RNN) model sequential information and are commonly used for the analysis of time series. The most usual operation to fuse information in RNNs is the sum. In this work, we use a RNN extended type, Long Short-Term Memory (LSTM) and we use it for image classification, to which we give a sequential interpretation. Since the data used may not be independent to each other, we modify the sum operator of an LSTM unit using the n-dimensional Choquet integral, which considers possible data coalitions. We compare our methods to those based on usual aggregation functions, using the datasets Fashion-MNIST and MNIST.**

***Index Terms*—Long Short-Term Memory, Recurrent Neural Network, Sequential Image Classification, Aggregation Functions, Choquet Integral.**

I. Introduction

Deep Neural Networks have proven to be a useful and accurate tool, positioning itself at the forefront of machine learning and pattern recognition [1], [2]. Deep Neural Networks have been applied in a multitude of applications, such as detection and segmentation in medical images [3], the prediction of financial values [4] or the prediction of words on the keyboard of the smartphones [5].

Recurrent Neural Networks (RNN) are a type of artificial neural network used for modeling sequential or temporal information, such as time series or Natural Language Processing (NLP) [4]–[6]. These networks consist of an architecture in which at each timestep, the output values of the layer of the previous instant are connected with the information of the current instant.

Classification is one of the most relevant problems in Machine Learning. The objective of a supervised classifier is to obtain intelligent knowledge from a set of labeled data where each data is assigned a class. Recurrent neural networks are usually used for classification based on strictly sequential information.

Besides, the information fusion process [7] is a fundamental process in many fields such as multi-criteria decision making [8], image processing [9], machine learning [10] or Convolutional Neural Networks (CNN) [11]. In the RNN, this information is stored in vectors, it has a multi-dimensional structure. To merge multivariate data, usually it is used the sum operator between the vectors as a form of aggregation of the sequential multivariate information.

However, there may be interaction between the recurrent data generated by the network and the data from the dataset. For this reason, we consider that it is convenient to use aggregation operators that take this fact into account. In this sense, in the aggregation functions literature fuzzy integrals have been used [12], which are based on fuzzy measures. These measures [13] allow us to take into account the relationship between the elements to be added, assessing the relevance of possible coalitions between the data [10]. One of the most widely used fuzzy integrals is the Choquet integral [14]. Until now, different generalizations of the Choquet integral [10], [15], [16], have been presented in the literature for one-dimensional data.

The objective of this work is to replace the classical aggregation operator used in the LSTM by the multidimensional Choquet integral, in such a way that

we generate a multidimensional information aggregation process in which the possible coalitions between the data are taken into account.

To show the usefulness of modifying the recurrent neural network aggregation process, we evaluate it in a classification problem. In particular, we are going to interpret the images as sequential information. We perform the sequential image classification using a Long Short-Term Memory (LSTM) [17], a widely used recurrent neural network. In the steps where the recurrent information is added to the initial information, we use the multidimensional version of the discrete Choquet integral.

The structure of this work is as follows. First, we discuss LSTM. In Section III, we explain the preliminary definitions and apply them to modify a LSTM. The experimental framework (used datasets and network architecture) is presented in Section IV. The results are showed and explained in Section V. Finally, some conclusions and future research are described in Section VI.

II. Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNN) [18] were born with the goal of modeling data with sequential or time dependence. Since neural network learning algorithms are based on the gradient, RNN may suffer from the vanishing gradient problem [19]. This problem lies in the recurrent decrease in the value of a variable at the output of the recurrent neural network. This is an especially serious problem when trying to train networks with long dependencies or time sequences, such as long time series or long texts.

The Long Short-Term Memory (LSTM) arises mainly as a response to this problem, representing a radical change in the recurrent neural network training because the information flow is regulated. The main idea of this architecture [17] is a memory cell based in gates which model the information enters or comes out.

Several modifications of the LSTM neurons have been considered in the literature [6], [20], [21]. In this work we are going to use one of the most extended [6].

A detailed representation of an LSTM unit can be seen in Fig. 1, where it is important the highlighting of the forget gate (f) [22], input gate (i), output gate (o)and candidate cell ($\tilde{c}$).

Next, we are going to explain the operation of the LSTM unit. Let N be the input sequence length, H the hidden size of the LSTM unit and T the number of timesteps. Then we get the following weights [6] for the matrices and vectors associated with the gates and the candidate cell:

- Input weight matrices: $\mathbf{W}_{fx}, \mathbf{W}_{ix}, \mathbf{W}_{cx}, \mathbf{W}_{ox} \in \mathbb{R}^{H \times N}$
- Recurrent weight matrices: $\mathbf{W}_{fh}, \mathbf{W}_{ih}, \mathbf{W}_{ch}, \mathbf{W}_{oh} \in \mathbb{R}^{H \times H}$
- Bias weight vectors: $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_c, \mathbf{b}_o \in \mathbb{R}^H$

The operations description for each timestep $t \in \{1, \ldots, T\}$ is the following:

i. The input values $\mathbf{x}^{(t)}$ and $\mathbf{h}^{(t-1)}$ enter to the gates f (Eq. 2), i (Eq. 3), $\tilde{c}$ (Eq. 4) and o (Eq. 6). In each of them, the value of $\mathbf{x}^{(t)}$ is multiplied by each of the input weight matrices ($\mathbf{W}_{gx}$, depending on the gate g). The same occurs with the values of $\mathbf{h}^{(t-1)}$ and the recurrent weight matrices. The H-dimensional vectors obtained from these multiplications with the corresponding bias $\mathbf{b}_g$ for each gate g are fused summing them.

ii. As activation function non-linear functions are used. As gate activation function the sigmoid logistic function (Eq. 1) is used:

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

As activation function of the candidate cell the hyperbolic tangent $\tanh(x)$ is used. Both of these functions are defined on $\mathbb{R}$. Here, on vectors are applied coordenate-wise.

iii. The previous timestep long-term memory vector ($\mathbf{c}^{(t-1)}$) and the candidate cell one ($\tilde{\mathbf{c}}^{(t)}$) are combined in this step. The Hadamard or element-wise product ($\circ$) is calculated between the values of the forget gate and input gate respectively (Eq. 5). Both values are added obtaining the current timestep value of the long-term vector ($\mathbf{c}^{(t)}$).

iv. Finally, the short-time memory vector ($\mathbf{h}^{(t)}$) is calculated. First, the long-term information ($\mathbf{c}^{(t)}$) is evaluated by the $\tanh(x)$ activation function. Subsequently, the Hadamard product is calculated between the value of the output gate ($\mathbf{o}^{(t)}$) and the information obtained from the last activation function, obtaining the value of the short-term memory vector ($\mathbf{h}^{(t)}$).

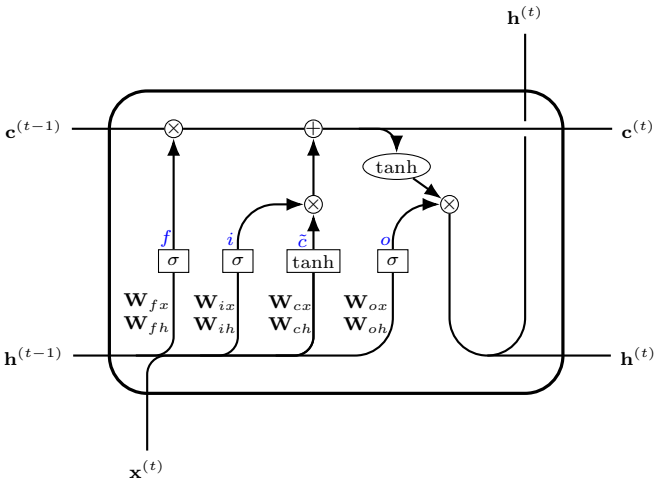

Fig. 1. LSTM unit representation

The equations that describe the explained process are the following (Eq. 2-7):

$$\mathbf{f}^{(t)} = \sigma(\mathbf{W}_{fx}\mathbf{x}^{(t)} + \mathbf{W}_{fh}\mathbf{h}^{(t-1)} + \mathbf{b}_f) \tag{2}$$

$$\mathbf{i}^{(t)} = \sigma(\mathbf{W}_{ix}\mathbf{x}^{(t)} + \mathbf{W}_{ih}\mathbf{h}^{(t-1)} + \mathbf{b}_i) \tag{3}$$

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}_{cx}\mathbf{x}^{(t)} + \mathbf{W}_{ch}\mathbf{h}^{(t-1)} + \mathbf{b}_c) \tag{4}$$

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \circ \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \circ \tilde{\mathbf{c}}^{(t)} \tag{5}$$

$$\mathbf{o}^{(t)} = \sigma(\mathbf{W}_{ox}\mathbf{x}^{(t)} + \mathbf{W}_{oh}\mathbf{h}^{(t-1)} + \mathbf{b}_o) \quad (6)$$

$$\mathbf{h}^{(t)} = \mathbf{o}^{(t)} \circ \tanh(\mathbf{c}^{(t)}) \quad (7)$$

III. Long Short-Term Memory modification

The objective of this section is to introduce a modification in a LSTM architecture. In the first subsection we define concepts as aggregation functions and the discrete Choquet-like integrals. In the second subsection we take these concepts to generalize the sum operation of the LSTMs, changing it.

A. Aggregation functions and Choquet-like integrals

For the fusion of the vectorial information in the recurrent neural networks as yet it is used the sum. The sum can be understood as an aggregation function. An aggregation function is a real-valued function which objective is the combination and merging of several values (usually numbers) in only one.

Taking a real interval $\mathbb{I} = [a, b] \subset \mathbb{R}$ which contents m values to aggregate ($x_1, \ldots, x_m \in [a, b]$), we can formally define an aggregation function as follows. A mapping $F : [a, b]^m \to [a, b]$ is called an aggregation function if satisfies:

1) Non-decreasing monotonicity: for all $(x_1, \ldots, x_m), (y_1, \ldots, y_m) \in [a, b]^m$, if $x_1 \leq y_1, \ldots, x_m \leq y_m$ then $F(x_1, \ldots, x_m) \leq F(y_1, \ldots, y_m)$
2) Boundary conditions: $F(a, \ldots, a) = a$ and $F(b, \ldots, b) = b$.

In this sense, the sum is an aggregation function on $\mathbb{R}$.

We denote the set formed by the natural numbers from 1 to m, $\{1, \ldots, m\}$ by M. An aggregation function is symmetric if for all $x_1, \ldots, x_m \in \mathbb{R}$ and for any permutation $\sigma : M \to M$ is fulfilled $F(x_1, \ldots, x_m) = F(x_{\sigma(1)}, \ldots, x_{\sigma(m)})$.

Let us denote by bold letters the elements in $\mathbb{R}^n$, that is, $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$. There is a partial order $\leq_P$ induced by $\leq$ order of real numbers, given as follows:

$$\mathbf{x} \leq_P \mathbf{y} \text{ if and only if } x_i \leq y_i$$

for all $i \in \{1, \ldots, n\}$.

The aggregation functions obtain a representative value from several inputs. It usually takes the data to be fused as independent variables. In many cases, the data to be fused has a correlation between them. In order to model correlation between the data, throughout the literature of the aggregation functions [7], [12], [23], [24] are used the fuzzy integrals. These integrals are based on fuzzy measures [13], [25] to model the possible coalition among the data. A function $\nu : 2^M \to [0, 1]$ is called a fuzzy measure on the M set if fulfills these two conditions:

1) Boundary conditions: $\nu(\emptyset) = 0$ and $\nu(M) = 1$
2) Monotonicity with respect to the inclusion: $\nu(A) \leq \nu(B)$ for every $A \subseteq B \subseteq M$

A fuzzy measure is called symmetric if the value of $\nu(A)$ depends only on the set cardinality, i.e., for all $A, B \subseteq M$, if $|A| = |B|$ then $\nu(A) = \nu(B)$.

A fuzzy measure considered in this work is the power measure, which is also a symmetric measure. It is defined for all $A \subseteq M$ as

$$\nu(A) = \left(\frac{|A|}{m}\right)^q \quad (8)$$

where $q > 0$ and $|A|$ is the cardinality of the set A. The q parameter allows the modeling of the interaction of the data. Throughout the literature, in the use of the power measure both fixed values and evolutionary methods [26] have been used to give value to the exponent q. In this work, the parameter is modeled by the stochastic gradient descent method, which is used in the learning process of the recurrent neural network obtaining most suitable q value [10].

Once the fuzzy measure is introduced, we present a concrete fuzzy integral: the discrete Choquet integral, which is also an example of aggregation function. The discrete Choquet integral [14], [27] with respect to the fuzzy measure $\nu : 2^M \to [0, 1]$ is defined as a mapping $Ch_\nu : \mathbb{R}^m \to \mathbb{R}$ such that

$$Ch_\nu(x_1, \ldots, x_m) = \sum_{i=1}^{m} \left(x_{\sigma(i)} - x_{\sigma(i-1)}\right) \nu\left(A_{\sigma(i)}\right) \quad (9)$$

where σ is a permutation on M where $x_{\sigma(1)} \leq \ldots \leq x_{\sigma(m)}$ with the convention $x_{\sigma(0)} = 0$ and $A_{\sigma(i)} := \{\sigma(i), \ldots, \sigma(m)\}$ is the subset of indices.

In many cases and concretely in the case of a recurrent neural network, the operations that are generally performed are using multidimensional structures like vectors. We take as an objective the aggregation of vectors in a single one, taking into account the modeling of the possible coalition between data. For this issue we are going to define the n-dimensional Choquet integral, which is a function that merges m vectors $\mathbf{x_1}, \ldots, \mathbf{x_m} \in \mathbb{R}^n$ in one.

Let m, n be positive integers. In this case, n represents the length (or dimensionality) of the vectors and m the number of vectors. Let $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_n)$ be a sequence of fuzzy measures on M and $Ch_{\nu_1}, \ldots, Ch_{\nu_n} : \mathbb{R}^m \to \mathbb{R}$ be Choquet integrals with respect to $\nu_1, \ldots, \nu_n$. A function $\mathbf{Ch}^r_{\boldsymbol{\nu}} : (\mathbb{R}^n)^m \to \mathbb{R}^n$ given by:

$$\begin{aligned}\mathbf{Ch}^r_{\boldsymbol{\nu}}(\mathbf{x_1}, \ldots, \mathbf{x_m}) = (&Ch_{\nu_1}(x_{11}, \ldots, x_{m1}), \ldots, \\ &Ch_{\nu_n}(x_{1n}, \ldots, x_{mn}))\end{aligned} \quad (10)$$

for all $\mathbf{x_1}, \ldots, \mathbf{x_m} \in \mathbb{R}^n$ is called a representable discrete Choquet-like integral with respect to $\boldsymbol{\nu}$. The Choquet-like integral $\mathbf{Ch}^r_{\boldsymbol{\nu}}$ is called representable since it is obtained by using n Choquet integrals on $\mathbb{R}$ separately for each component (Eq. 10, Fig. 5). As we can see, this expression is a generalization of the standard Choquet integral, since the input vectors are n-tuples with the same coordinates, i.e. $\mathbf{x} = (x, \ldots, x)$ and $\nu_1 = \ldots = \nu_n = \nu$, the output is an n-tuple with the same coordinates equal to the input of Ch_ν.

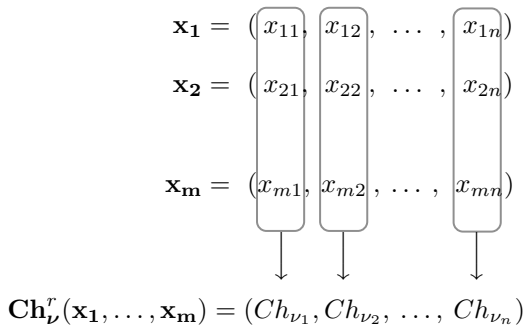

Fig. 2. Graphical representation of the representable discrete Choquet-like integral

B. LSTM modification based on aggregation functions and representable Choquet integral

The objective of the present section is the application of the definitions explained in the previous one in order to modify and improve the LSTM architecture. In this sense, we modify the aggregation operator of the LSTM network (vector summation) for the representable Choquet integral.
With this new approach, Eqs. 2-7 are transformed into the following set of equations (Eqs. 11-16) to describe the process:

$$\mathbf{f}^{(t)} = \sigma\left(\mathbf{Ch}^r_{\boldsymbol{\nu}}(\mathbf{W}_{fx}\mathbf{x}^{(t)}, \mathbf{W}_{fh}\mathbf{h}^{(t-1)}, \mathbf{b}_f)\right) \tag{11}$$

$$\mathbf{i}^{(t)} = \sigma\left(\mathbf{Ch}^r_{\boldsymbol{\nu}}(\mathbf{W}_{ix}\mathbf{x}^{(t)}, \mathbf{W}_{ih}\mathbf{h}^{(t-1)}, \mathbf{b}_i)\right) \tag{12}$$

$$\tilde{\mathbf{c}}^{(t)} = \tanh\left(\mathbf{Ch}^r_{\boldsymbol{\nu}}(\mathbf{W}_{cx}\mathbf{x}^{(t)}, \mathbf{W}_{ch}\mathbf{h}^{(t-1)}, \mathbf{b}_c)\right) \tag{13}$$

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \circ \mathbf{c}^{(\mathbf{t-1})} + \mathbf{i}^{(t)} \circ \tilde{\mathbf{c}}^{(t)} \tag{14}$$

$$\mathbf{o}^{(t)} = \sigma\left(\mathbf{Ch}^r_{\boldsymbol{\nu}}(\mathbf{W}_{ox}\mathbf{x}^{(t)}, \mathbf{W}_{oh}\mathbf{h}^{(t-1)}, \mathbf{b}_o)\right) \tag{15}$$

$$\mathbf{h}^{(t)} = \mathbf{o}^{(t)} \circ \tanh(\mathbf{c}^{(t)}) \tag{16}$$

As we have shown before, the main modification in the unit performance is the replacement of the sum operation for $\mathbf{Ch}^r_{\boldsymbol{\nu}}$. Nevertheless, for completeness in the study, in the fusion of multidimensional vectorial information in the LSTM unit we will use different aggregation functions. The aggregation functions we are going to use will be the following:

- Maximum function
- $\mathbf{Ch}^r_{\boldsymbol{\nu}}$ using ν as power measure (ν_q, Eq. 8). In that sense we will use q in two cases:
  - With a fixed value , $q = 2$
  - Learning $q \in \mathbb{R}^+$ with the stochastic gradient descent in the neural network
- Sum operation, to compare with the rest of the operators.

IV. Experimental framework

In the present section we explain the datasets, neural network architecture performance as well as the selected hyperparameters and optimizer.

A. Datasets

The datasets used in this work are the following:

- *Fashion-MNIST (F-MNIST)* [28]. It consists of a training set of 60,000 images of dimensions 28×28 distributed in 10 classes. The test set consists of similar 10,000 images. Both sets are balanced. The images correspond to 10 different items from clothing categories .
- *MNIST* [29]. It consists of a training set of 60,000 images of dimensions 28×28 distributed in 10 classes. The test set consists of similar 10,000 images. Both sets are balanced. It is a subset of the NIST dataset and consists of a handwritten 0 to 9 digit image dataset.

Both *F-MNIST* and *MNIST* datasets have been extensively used in benchmarks, in pattern and isolated digit handwriting classification [30] respectively. In this case, we consider the data contained in an image as sequential information [31].
We transform both *F-MNIST* and *MNIST* input values into a sequence classification problem by scanning the images from up to down using horizontal slices of the image [30].

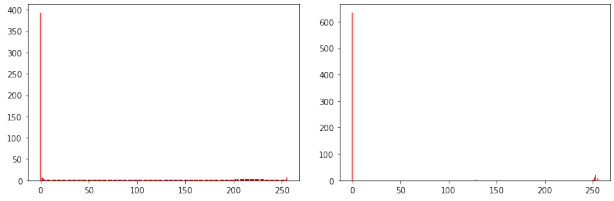

Fig. 3. F-MNIST and MNIST dataset average histograms

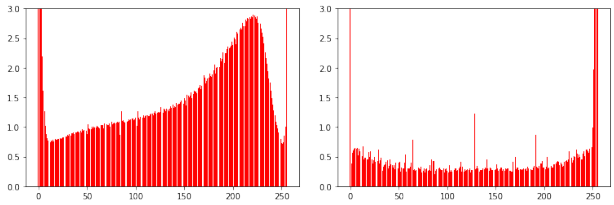

Fig. 4. F-MNIST and MNIST dataset average histograms between 0 and 3

In Fig 3 we show the mean histograms of all the images of the *F-MNIST* and *MNIST* datasets respectively. However, as the value 0 predominates well above the rest, in Fig 4 we show the histogram but limiting the y-axis to 3. In this case, we can see the difference in the distribution of the pixels of the images of both datasets. We can observe that in the case of the *F-MNIST* dataset there may be more distributed information, but there may also be greater correlation and interaction between the data.

B. Architecture

In this case, as we can see in Fig. 5, we set an architecture where the images are taken as sequential data. In each timestep $t \in \{1, \ldots, T\}$ a row of the image is taken as a vector. This vector is the input data $\mathbf{x}^{(t)} \in [0, 1]^N$, and in the used dataset concretely, it holds $T = N = 28$.
The used architecture (Fig. 5) consists in two layers:

- *LSTM unit* (Section II and III-B): With hidden size fixed in H. We have done the experiment with different sizes for this parameter. To test the performance of our new way of fusing vectorial data in LSTM, we have fixed to $H = 32$, $H = 64$ and $H = 128$.
- *Fully connected layer*: In second place, we have used a fully connected layer which connects the H nodes of the LSTM unit with the number of classes of nodes, in this case, 10. A probability value in $[0, 1]$ is assigned to each of them. It is classified in the class number corresponding to the maximum probability value of the vector.

In this experiment 10 independent runs of 40 epochs each have been executed. The fixed learning rate for it has been $\alpha = 0.1$ and the optimization method for the learning has been the Stochastic Gradient Descent (SGD) [32]. The used loss function has been the Cross-Entropy Loss function [33], given by the following expression (Eq. 17):

$$\mathcal{L}(y) = \frac{1}{|C|} \sum_{p=0}^{|C|-1} -\log\left(\frac{e^{y_p}}{\sum_{j=0}^{|C|-1} e^{y_j}}\right) \tag{17}$$

where y corresponds to the real values of the dataset, C is the set of the classes, $|C|$ the number of classes and p the predicted class.

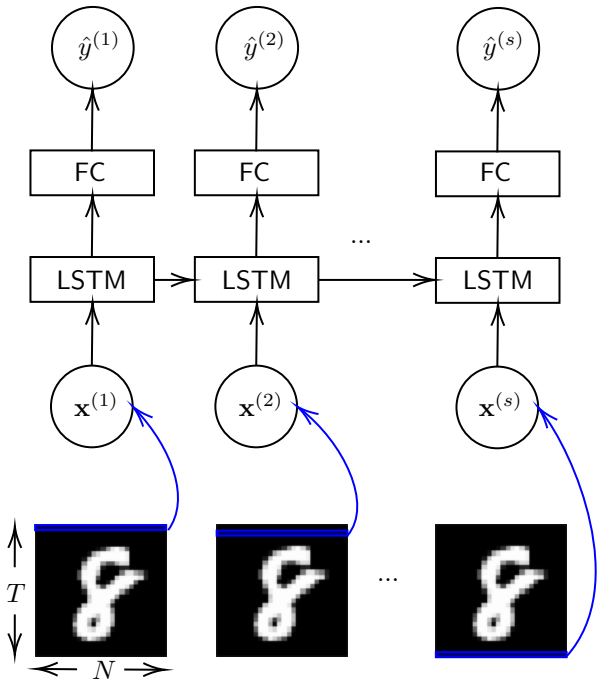

Fig. 5. Graphical representation of the used network architecture

V. Results

In this section, we present the obtained accuracy results for the discrete multidimensional Choquet integral performance in LSTM units. In Table I the average results done for 10 independent runs for 40 epochs for each aggregation function are shown.

In Table I the results for each aggregation function as well as for each hidden size and dataset are shown.

In the *F-MNIST* case, all the best results are obtained when we aggregate the values through the Choquet integral, but when the exponent $q > 0$ is learned by the recurrent neural network itself. This means that when the q-learned Choquet integral is used, the algorithm models better the interaction and the possible coalition between the data. In this way, when the hidden size of the LSTM unit is fixed to 128, we obtain a performance that allows a data weighting which improves 1.01 accuracy points with the respect to the classical form of aggregation in this architecture, the sum. In the case of smaller hidden sizes (64 and 32) the average performance is better than the sum one, but only 0.17 and 0.21 accuracy points better, respectively.

On the other hand, with the respect to the *MNIST* dataset, the improvement is much less, where fixing a hidden size of 128 units is 0.03 points better. One reason to justify this non-improvement may be the fact that the data are less correlated with each other and there is less interaction between them as we have seen in Fig. 4 and Section IV.

VI. Conclussion

In this work we have proposed the use of a new method for the fusion of multidimensional vectors, in the process of fusion of sequential information in a concrete type of the recurrent neural networks. Likewise, an improvement in precision in the experiments carried out has been corroborated. We have observed that better results are obtained when we substitute the sum for the Choquet integral.

Regarding future lines of research, in the theoretical aspect our intention is to continue investigating new forms of fusion of vectors based on the Choquet integral, such as the generalization of expressions. On the applied side, future lines go in the direction of modifying more complex architectures, as well as the use of these architectures of other types of problems (sentiment analysis, prediction of time series, etc.)

Acknowledgement

This work has been funded by the Agencia Estatal de Investigación (España) under the project PID2019-108392GB-I00 (AEI/10.13039/ 501100011033), by the Immigration Policy and Justice Department of the Government of Navarre-Tracasa Instrumental and the Grant VEGA 1/0267/21 (Slovakia).

References

[1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, p. 85–117, Jan 2015.

[2] L. Shiloh-Perl and R. Giryes, "Introduction to deep learning," 2020.

[3] T. Lei, R. Wang, Y. Wan, B. Zhang, H. Meng, and A. K. Nandi, "Medical image segmentation using deep learning: A survey," 2020.

[4] Q. Tang, T. Fan, R. Shi, J. Huang, and Y. Ma, "Prediction of financial time series using lstm and data denoising methods," 2021.

[5] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions," 2018.

[6] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

TABLE I
AVERAGE ACCURACY RESULTS FOR DIFFERENT AGGREGATION FUNCTIONS AND HIDDEN SIZES

	F-MNIST						MNIST					
	h=32		h=64		h=128		h=32		h=64		h=128	
Aggregation	Acc.	SD	Acc.	SD	Acc.	SD	Acc.	SD	Acc.	SD	Acc.	SD
max	88.45	0.22	89.04	0.95	86.45	0.69	98.17	0.10	98.69	0.09	98.88	0.10
Ch_2	85.45	0.00	86.25	0.87	85.65	0.56	97.95	0.18	98.66	0.16	98.88	0.08
Ch_q	**88.68**	0.25	**89.34**	0.23	**90.01**	0.18	**98.23**	0.05	98.77	0.08	**98.93**	0.06
$\sum$	88.47	0.35	89.17	0.26	89.00	0.23	98.19	0.12	**98.80**	0.11	98.90	0.13

[7] G. Beliakov, A. Pradera, and T. Calvo, "Aggregation functions: A guide for practitioners," in *Studies in Fuzziness and Soft Computing*, 2007.

[8] L. De Miguel, M. Sesma-Sara, M. Elkano, M. Asiain, and H. Bustince, "An algorithm for group decision making using n-dimensional fuzzy sets, admissible orders and owa operators," *Information Fusion*, vol. 37, pp. 126–131, 2017.

[9] D. Paternain, J. Fernandez, H. Bustince, R. Mesiar, and G. Beliakov, "Construction of image reduction operators using averaging aggregation functions," *Fuzzy Sets and Systems*, vol. 261, pp. 87–111, 2015. Theme: Aggregation operators.

[10] G. Lucca, J. A. Sanz, G. P. Dimuro, B. Bedregal, M. J. Asiain, M. Elkano, and H. Bustince, "Cc-integrals: Choquet-like copula-based aggregation functions and its application in fuzzy rule-based classification systems," *Knowledge-Based Systems*, vol. 119, pp. 32–43, 2017.

[11] C. A. Dias, J. C. S. Bueno, E. N. Borges, S. Botelho, G. Dimuro, G. Lucca, J. Fernández, H. Bustince, and P. Drews, "Using the choquet integral in the pooling layer in deep learning networks," in *NAFIPS*, 2018.

[12] M. Grabisch and C. Labreuche, "A decade of application of the choquet and sugeno integrals in multi-criteria decision aid," *Annals of Operations Research*, vol. 175, pp. 247–286, 2008.

[13] G. Lucca, J. A. Sanz, G. P. Dimuro, E. N. Borges, H. Santos, and H. Bustince, "Analyzing the performance of different fuzzy measures with generalizations of the choquet integral in classification problems," in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6, 2019.

[14] G. Choquet, "Theory of capacities," *Annales de l'Institut Fourier*, vol. 5, pp. 131–295, 1954.

[15] H. Bustince, R. Mesiar, J. Fernandez, M. Galar, D. Paternain, A. Altalhi, G. Dimuro, B. Bedregal, and Z. Takáč, "d-choquet integrals: Choquet integrals based on dissimilarities," *Fuzzy Sets and Systems*, vol. 414, pp. 1–27, 2021. Aggregation Functions.

[16] G. Lucca, J. Antonio Sanz, G. P. Dimuro, B. Bedregal, H. Bustince, and R. Mesiar, "Cf-integrals: A new family of pre-aggregation functions with application to fuzzy rule-based classification systems," *Information Sciences*, vol. 435, pp. 94–110, 2018.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] D. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[19] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks* (S. C. Kremer and J. F. Kolen, eds.), IEEE Press, 2001.

[20] F. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, vol. 3, pp. 189–194 vol.3, 2000.

[21] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005. IJCNN 2005.

[22] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, pp. 850–855 vol.2, 1999.

[23] M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap, *Aggregation Functions*. Encyclopedia of Mathematics and its Applications, Cambridge University Press, 2009.

[24] M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap, "Aggregation functions: Means," *Inf. Sci.*, vol. 181, pp. 1–22, 2011.

[25] T. Murofushi, M. Sugeno, and M. Machida, "Non-monotonic fuzzy measures and the choquet integral," *Fuzzy Sets and Systems*, vol. 64, no. 1, pp. 73–86, 1994.

[26] E. Barrenechea, H. Bustince, J. Fernandez, D. Paternain, and J. A. Sanz, "Using the choquet integral in the fuzzy reasoning method of fuzzy rule-based classification systems," *Axioms*, vol. 2, no. 2, pp. 208–223, 2013.

[27] L. Jin, M. Kalina, R. Mesiar, and S. Borkotokey, "Discrete choquet integrals for riemann integrable inputs with some applications," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 3164–3169, 2018.

[28] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *ArXiv*, vol. abs/1708.07747, 2017.

[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] T. M. Breuel, "Benchmarking of lstm networks," 2015.

[31] Q. V. Le, N. Jaitly, and G. E. Hinton, "A simple way to initialize recurrent networks of rectified linear units," *CoRR*, vol. abs/1504.00941, 2015.

[32] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Advances in Neural Information Processing Systems 20 (NIPS 2007)* (J. Platt, D. Koller, Y. Singer, and S. Roweis, eds.), pp. 161–168, NIPS Foundation (http://books.nips.cc), 2008.

[33] K. Murphy, "Machine learning - a probabilistic perspective," in *Adaptive computation and machine learning series*, 2012.