Recursive neural networks: recent results and applications

Andreas Zelios[1], Achilleas Grammenos[1], Maria Papatsimouli[1], Nikolaos Asimopoulos[1], and George Fragulis[1]
[1] Department of Electrical and Computer Engineering, University of Western Macedonia, 501 00 Kozani, Greece

Abstract. Neural Network's basic principles and functions are based on the nervous system of living organisms, they aim to simulate neurons of the human brain to solve complicated real-world problems by working in a forward-only manner. A recursive Neural Network on the other hand is based on a recursive design principle over a given sequence input, to come up with a scalar assessment of the structured input. This means that is ideal for a given sequence of input data that is when processed dependent on its previous input sequence, which by default are used in various problems of our era. A common example could be devices such as Amazon Alexa, which uses speech recognition i.e., given an audio input source that receives audio signals, tries to predict logical expressions extracted from its different audio segments to form complete sentences. But RNNs do not come with no problems or difficulties. Today's problems become more and more complex involving parameters in big data form, therefore a need for bigger and deeper RNNs is being created. This paper aims to explore these problems and ways to reduce them while also providing a description of RNN's beneficial nature and listing different uses of the state-of-the-art RNNs and their use in different problems as those mentioned above.

1 Introduction

ML can be described as a subset of artificial intelligence in which a mathematical model based on "training data" is built to make decisions or predictions without being programmed [44]. It has been applied for several scientific purposes, such as education applications [50], sign language learning [45], gaming applications [46-47], and early and objective autism spectrum disorder (ASD) assessment [48-49] and various other applications [51-53]. Feedforward Neural Networks (FNNs) is used on a great scale in numerous fields to solve problems and create high-importance applications. Despite the many advantages, FNNs have a significant drawback: they have any kind of memory. This feature can cause multiple difficulties in some applications such as Statistical language modeling [1] and so on. The solution to this problem is the Recurrent Neural Networks (RNNs) which use feedback. RNNs use results of shallower layers to 'feed' the deeper ones. In other words, they include loops that assist in the usage of previously established knowledge as appears in fig. **1**. The activation functions that are usually utilized are Sigmoid function, Tanh function, and Relu function, but any other desired function can be used as well. The training process is usually fulfilled with the aid of the Backpropagation through time algorithm (BPTT) [2]. The BPTT algorithm is "an extension of the backpropagation algorithm for recurrent networks". ". It unfolds the neural network through time and as it seems in fig. **2** and then it makes use of the regular backpropagation algorithm. There are four types of RNNs: one-to-one, one-to-many, many-to-one, many-to-many (fig. **3**) [3]. The most commonly used type is one-to-one, which

has one input and one output. A typical example is image classification [4], where the input is one image and the output is the category that includes this image. One-to-many uses one input and exports multiple outputs.

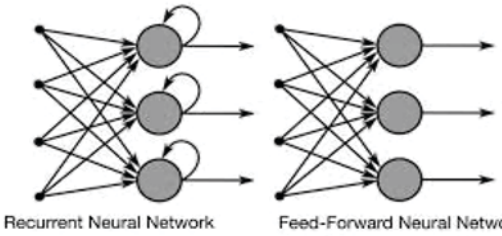

Fig 1. RNN and FFN

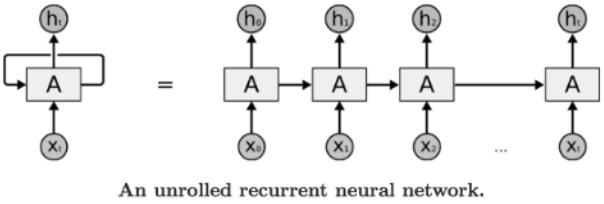

Fig 2. BPTT

Such application could be multilingual speech translation [5]. An example of the many-to-one type is Mobility Prediction at Points of Interest [6]. RNNs have many features that make them attractive and useful.

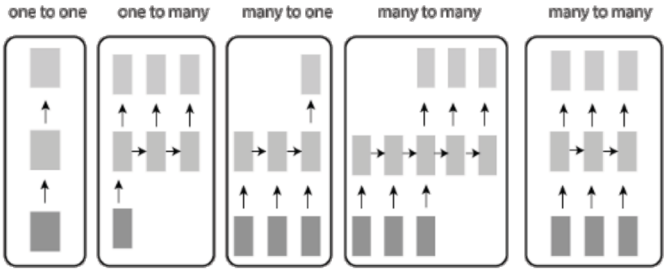

Fig 3. RNN Types

First of all, as mentioned above, they can support various sizes of inputs and outputs, which makes them suitable for a wide range of applications. Another advantage is feedback utilization which increases the efficiency of the neural network. Furthermore, RNNs are desirable for sequence data [7]. Despite all their capabilities, RNNs have some weaknesses. One of them is the vanishing gradient problem [8], which appears in the FNNs as well and it creates difficulties in the training process. A solution to this problem would be Long short-term memory (LSTM) [9].

2 Uses of recurrent networks in today's world

2.1 Automatic speech recognition

Currently, there are three methods for speech recognition in RNN [10][11][12][13].

- Connectionist Temporal Classification (CTC)
- Attention Encoder-Decoder (AED)
- RNN Transducer (RNN-T)

All the above are based on end-to-end systems that require a single network architecture in a deep learning framework, in comparison to traditional techniques like Hidden Markov Models (HMM), Gaussian Mixture Models (GMM) in combination with other modules, followed by training which also requires a language model [14]. Traditional methods also need integration into a Weighted Finite-State Transducer (WFST) for efficient decoding making the process of creating these networks hard for non-experts in language translation and machine learning fields. RNN in ASR outputs a sequence of (Characters + 1) character probabilities per sequence element, a blank character "-" is also added to the output. Each time step the RNN is picking a character and fuses them to form a path P. To encode a character, we observe one or multiply adjacent characters like the one that we choose on the path that is followed by one or more black characters.

- CTC in RNNs is mainly used as a loss function after training [15]. It outputs a matrix containing character probabilities for each time-step from a reduced input sequence, then a decoder uses those probabilities and maps them to the final text (labeling) [16][17].
- On the other hand, AED models consist of an encoder, an Attender, and a decoder that contains the RNN properties. AED also doesn't have repeated or "-" characters to interpose the final predictions [18]. Initially, the input is encoded to a sequence of high feature vectors H which is smaller than the initially given input as in the CTC model, by a reduction factor. Then the output is passed to a decoder that processes the input to a predicted output in a recurrent state manner.
- RNN Transducer is the least explores of the three architectures. It consists of three elements [19], the RNN transducer which outputs a probability sequence for the input, a transcription network (bidirectional RNN) that scans the input and outputs a sequence of transcription vectors, and a Prediction Network (RNN with a single hidden layer) that models each element of the input sequence given the previous ones. Memory cost during training is much higher than the previous two architectures [20], although attempts to minimize this cost are made recently.

2.2 Video Tagging

In this category a video is tagged about its context, this can be applied to hashtags of the videos in the description for users who want to search this subject or to validate if the context of a video is faithful to its promise. [21] has used RNNs for clickbait Identification. The title and the description were fed into a deep neural network after some pre-processing in an Embedding Layer i.e., a layer that creates word embeddings from the text. The output is then passed through a fully connected layer that outputs the results. [22] Uses LSTM to compute the significance of given context words to a target in a sentence. In comparison to SVM and the lexicon-enhanced neural network, it provided state-of-the-art results. [23] Then used a

similar method to extract the opinion of reviewers from review videos using a two-pass bidirectional RNN architecture with the addition of an attention component.

2.3 Deepfake detection

Deepfake detection is the process of finding if a face of an individual was replaced by the face of another individual. Many methods were proposed for this problem. [24] Focused on the feature of Eye blinking. Long-Term Recurrent CNN was used because eye blinking is a feature with high temporal dependencies. The network was used for feature extraction, sequence learning, and state prediction. LSTM-RNN was also incorporated to minimize the gradient vanishing problem and remove large memory usage as a problem. On the other hand [25] extracts features at the frame level not just of the blinking eye using a CNN and compares it to an RNN for temporal inconsistencies. A SoftMax layer is then used to determine whether the frame is authentic or deepfake.

3 Advantages – disadvantages of using recurrent networks and ways to reduce the drawbacks

3.1 Advantages

The main feature that distinguishes the RNNs from the FNNs, is memory. Thanks to this feature, RNNs make use of previously established knowledge, which allows them to make more accurate predictions. For this reason, RNNs are suitable for problems containing sequential data like time series, speech, text, financial data, audio, video, weather, and much more. Some applications that take advantage of the ability of RNNs to handle sequential data are Natural Language Processing (NLP) [26] where a series of words are being processed one by one in word series, speech recognition [27] where audio is transformed into word series and other domains such as the ones mentioned in the above section.

3.2 Disadvantages – ways to solve them

3.2.1 Slow training time

The sequential nature of the problems that RNN solve produce high temporal dependencies. The training process can use parallelization [28] by batching or grouping the sequences and distributing them in the computations. Another method that is much more commonly used is static or dynamic subsampling where the length of the sequence is reduced by a factor as explained in the AED models, however, this method produces much redundant information and may avoid critical information. State-of-the-art results [14] are still produced [28], but a combination with Highway Networks is advised [29][30]. Those networks decide the amount of information to be carried to the next layer a min operator is set to ensure a valid probability. SOUL [31] networks are another way of reducing the training time by Limiting the size of the vocabulary in statistical language modeling. It's a hierarchical probabilistic network but is constructed by a clustering tree so that a class can be a member of another class. So a sequence c1:Depth(wi) = c1,…,cd encodes the path for the word ith word w_i in the clustering tree. Where $c_d(w_i)$ is a class or subclass of the w_i word and $c_d(w_i)$ is the leaf of w_i. A probability is then formed as:

$$P(w_i|h) = P(c_1(w_i)|h) \prod_{d=2}^{D} P(c_d(w_i)|h, c_{1:d-1})$$

A SoftMax function at each level is used, so each word ends up forming its class in some leaf of the tree. The overall network can be seen in the bellow figure provided by Le et al. (2011) [31] (fig. **4**).

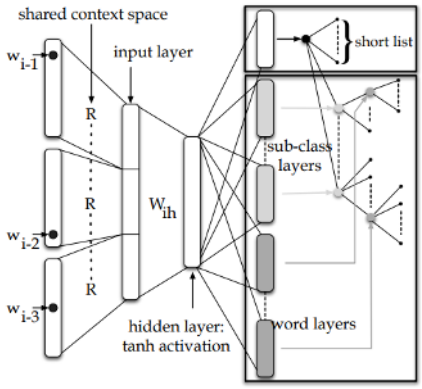

Fig 4. SOUL Networks architecture.

3.2.2 Fixed amount of hidden neurons

There are two methods to increase the number of hidden neurons in an RNN architecture. The first is adding hidden neurons during training [32], the most basic algorithm is the Cascade Correlation Network where the output from all neurons are fed into the newly added neurons. A learning algorithm attempts to maximize the amount of association between the new neuron's output and a minimum error. The second method starts with a large network that is constructed by a large number of hidden neurons and gradually reduces the amount by removing weights and neurons that are considered less useful this method is referred to as based compressed RNN [33][34].

3.2.3 Small context size in practice due to the vanishing gradient problem

To minimize the vanishing gradient problem in RNNs many methods have been found with the most popular being the use of Long short-term memory modules. It extends the basic RNN structure by adding a memory block in the place of the hidden unit that has an input and output gate. The input gate controls the activation of inputs, if the value of activation is near zero the activation of the cell will not be overwritten by the new inputs and may be provided by the output much later in the sequence. A basic memory block can be seen below figure provided by Hasim et al. [35] (fig. **5**).

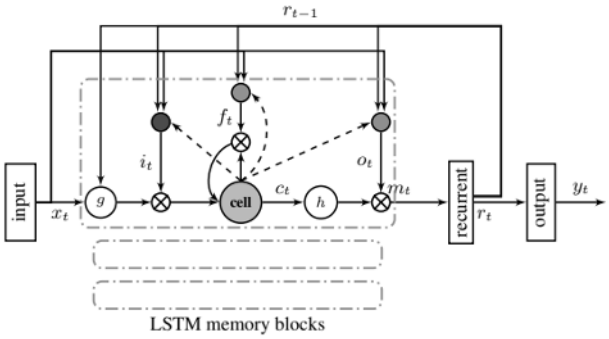

Fig 5. LSTM memory Block.

Bidirectional neural networks are also a way to reduce the vanishing gradient problem. It includes the context contained in previous training examples by adding an extra hidden layer (backward hidden layer) that is not connected with the original hidden layer but only with the output layer. [36] Uses this kind of network in a recent study to Improve protein disorder prediction achieving a much greater ROC curve that the other methods used.

4 State-of-the-art implementations of recurrent networks

4.1 Signal temporal logic (STL)

As was previously mentioned RNNs do have some drawbacks, one of them being the uncertainty in the quality of results produced in large-scale system predictions such as the prediction of the weather. STLnet [37] aims to solve this specific problem by incorporating Signal temporal logic into the already existing RNN models. STL specifies temporal properties of discrete and continuous signals [38] and uses them as system constraints of the problem being solved, in this manner STLnet specifies temporal properties of RNN output signals which are discrete-time signals. Properties that are specified are:

- Reasonable Range: A constraint in the range of the sequence value should be made to be reasonable. The potential interdependence of variables as well as the variety of the range between variables makes the constraint of the utmost importance in RNNs.
- Consecutive Changes: The consecutive changes in a specific period should follow the model properties.
- Resource Constraint: The prediction results have to take into account the available resources of the problem. An accurate and representative definition of the right result according to these resources should be made. For example, efficiency of an electrical plant cannot be 100% or greater.
- Variable and Temporal Correlation: The constraint that is created due to the dependencies between different variables and locations overtime.
- Existence: An effort to pinpoint the importance of a variable to the predicted outcome should be made, so that unnecessary variable won't be included.
- Unusual Cases: Uncertainties highly affect the predicted outcome because they limit the number of instances provided. For instance, on Black Friday, the social mobility in the

shops is expected to be greater than on other days. If possible, a specification of the properties of those cases should be included in the training of the model.

Following the compression of knowledge into a single model [39], STLNet uses two networks a teacher and a student network. As its name implies teacher network assists the student network when fails to predict a sequence that follows the model properties by generating a close to the produced sequence that satisfies the model properties. The student network then uses the target trace and the teacher network's output to update its parameters and get closer to the desired outcome output. In the tables below, we can see the comparison between Long Short-Term Memory and Transformer RNNs with and without STLnet teacher and student networks [37].

	LSTM			LSTM STLnet-p			LSTM STLnet-q		
	RMSE	Sat Rate	Violate ϱ	RMSE	Sat Rate	Violate ϱ	RMSE	Sat Rate	Violate ϱ
ϕ1	0.026	92.00%	-0.298	0.025	98.34%	-0.014	0.025	100%	0
ϕ2	94.304	75.61%	-117.982	90.016	97.78%	-1.603	90.160	100%	0
ϕ3	4.214	75.47%	-1.589	4.209	87.69%	-0.606	4.209	100%	0
ϕ4	0.309	56.68%	-36.884	0.230	83.09%	-3.906	0.229	100%	0
ϕ5	2.118	0.84%	-463.534	1.151	75.64%	-19.842	1.162	100%	0
ϕ6	8.603	59.54%	-282.404	8.532	61.85%	-282.403	7.122	100%	0

Table 1. LSTM with and without STLnet

	Transformer			Transformer STLnet-p			Transformer STLnet-q		
	RMSE	Sat Rate	Violate ϱ	RMSE	Sat Rate	Violate ϱ	RMSE	Sat Rate	Violate ϱ
ϕ1	0.045	27.76%	-18.808	0.031	89.48%	-1.835	0.031	100%	0
ϕ2	105.211	49.44%	-109.282	111.688	76.08%	-18.874	111.655	100%	0
ϕ3	4.340	52.96%	-3.855	4.339	60.70%	-2.596	4.339	100%	0
ϕ4	0.124	0.36%	-38.893	0.135	51.00%	-5.101	0.135	100%	0
ϕ5	2.196	8.88%	-31.172	1.805	50.20%	-4.612	1.804	100%	0
ϕ6	8.156	20.08%	-301.175	8.326	20.32%	-307.165	2.657	100%	0

Table 2. Transformer with and without STLnet

We can note that in ϕ6 unusual cases were included, we can see that the addition of STL greatly improves results lowering the root mean square error and improving the satisfaction rate. The authors also tested the STLNet in a real-world problem by training RNN LSTM models with air quality datasets which include pollutants quantities and meteorological readings. To build the LSTM network, they regarded one pollutant as one variable and concatenated all variables from the same time unit. Next, they specified some of the above-mentioned model properties, including reasonable ranges, consecutive changes, correlations between different pollutants, and between different locations, etc. As we can see in the figure below (fig. **6**) in the first graph (a), for a small number of prediction lengths, there is any significant difference between LSTM with and without STLNet addition. But as the prediction length increases, it becomes clear, that STLNet helps in the error decrease.

In graph (b), the satisfaction rate is shown to be much higher in all prediction lengths. Finally, in the last two graphs (c), (d) we can see its use in missing data cases or unusual cases. We can conclude that again the LSTM and STLNet combination outperforms the plain LSTM RNN network.

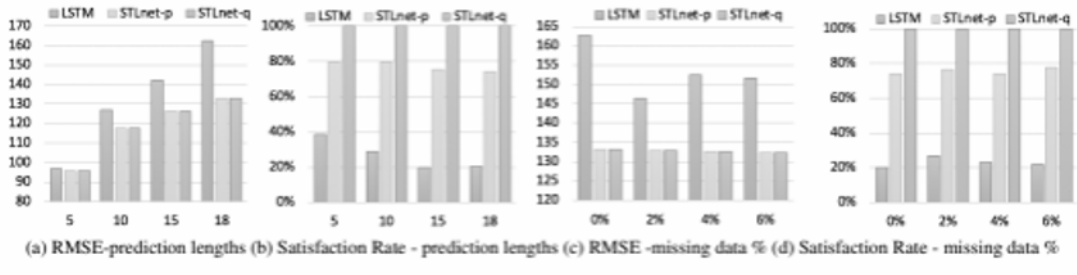

Fig 6. Addition of STLnet for air quality prediction.

4.2 Dual-path RNN

Conventional RNNs do not specialize in processing long data sequences, although it is necessary in many cases. Besides that, RNNs can earn this capability with the aid of Dual-Path RNN (DPRNN) [41]. DPRNN, organizes RNN layers to produce a smaller description of a model in three phases provided visually in (fig. **7**).

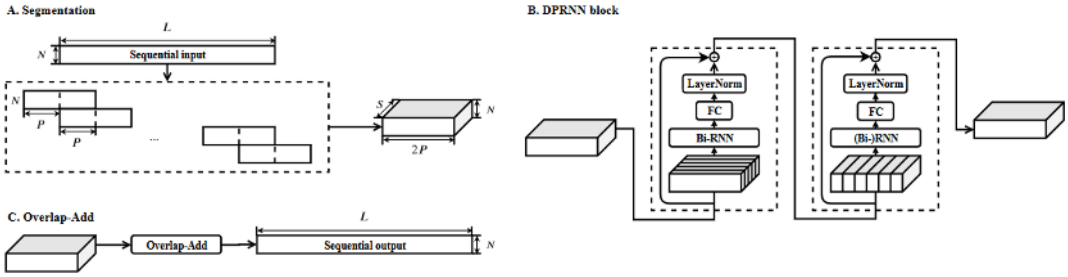

Fig 7. DPRNN architecture

The first phase is segmentation in which the sequential input is divided into smaller parts and concatenated them to form a 3-D tensor. The second phase is block processing, which is composed of two modules: intra-chunk processing and inter-chunk processing. The intra-chunk includes bi-directional RNN layers which process local information of the 3-D tensor, and the inter-chunk uses RNN layers to process global information. In both modules, the RNNs are followed by a fully-connected layer and a normalization layer to balance out the result. Finally, the overlap-add stage is applied to transform the output (which is in 3-D form) into sequential form. To evaluate the method's performance, the authors tested it on the time-domain audio separation network (TasNet) with different separation networks (Table **3**) [42]. Specifically, they used the WSJ0-2mix dataset [43] which contains two-speaker speech

separation data. The utilized metrics were the Signal-to-Noise Ratio improvement (SI-SNRi), which compares the level of the desired signal to the level of background noise, and the Signal-to-Distortion Ratio improvement (SDRi), which measures the decrease of the distortion and the model size. Compared with CNN-based architectures such as temporal convolutional networks (TCNs), DPRNNs 'achieve superior performance with an even smaller model size'. They also compared previously used methods to DPRNN (Table **4**) concluding with the same result, a much smaller model size (2.6 M) and similar to other method results (SI-SNRi of 18.8 dB and SDRi of 19 dB).

Separator network	Model size	Window (samples)	Chunk size (frames)	SI-SNRi (dB)	SDRi (dB)
TCN	5.1M	16	–	15.2	15.5
DPRNN	**2.6M**	16	100	16.0	16.2
		8	150	17.0	17.3
		4	200	17.9	18.1
		2	250	**18.8**	**19.0**

Table 3. Comparison between TCN-TasNet and DPRNN-TasNet

Method	Model size	SI-SNRi (dB)	SDRi (dB)
DPCL++ [31]	13.6M	10.8	–
uPIT-BLSTM-ST [29]	92.7M	–	10.0
ADANet [32]	9.1M	10.4	10.8
WA-MISI-5 [33]	32.9M	12.6	13.1
Conv-TasNet-gLN [4]	5.1M	15.3	15.6
Sign Prediction Net [34]	55.2M	15.3	15.6
Deep CASA [20]	12.8M	17.7	18.0
FurcaNeXt [5]	51.4M	–	18.4
DPRNN-TasNet	**2.6M**	**18.8**	**19.0**

Table 4. Previously used methods compared to DPRNN

A hybrid system was also created to show the efficiency of the network in speech separation as well as recognition. The comparison was set under a single-speaker noisy environment, once again between TCN and DPRNN TasNet models (Table **5**).

Separator network	Model size	SI-SNRi (dB)	WER (%)
TCN	5.1M	7.6	28.7
DPRNN	**2.6M**	**8.4**	**25.9**
Noise-free reverberant speech	–	–	9.1

Table 5. Speech recognition and separation comparisons of DPRNN and TCN

Results indicate that the addition of DPRNN concludes in much greater results with smaller resource usage.

5 Conclusions

Recurrent Neural Networks are being used for more than thirty years in various applications of our everyday lives including our two main senses, audio (speech recognition-production) and vision (OCR, video tagging, Deepfake detection). Even though other methods and

technologies have appeared in sequential data processing, RNNs with the aid of some small extensions remain powerful and extremely useful as the above-mentioned methods and experiments indicated.

References

1. W. D. Muldera, S. Bethardb, M. F. Moens, Computer Speech & Language, **30** (2015)
2. T.Mikolov, S. Kombrink, L. Burget, J. Černocký, S. Khudanpur, ICASSP, 22-27 May 2011, Prague, Czech Republic (2011)
3. N. K. Manaswi, Deep Learning with Applications Using Python, 115-126 (2018)
4. M. I. Lakhal, H. Çevikalp, S. Escalera, F. Ofli, IET Computer Vision, **12** (2018)
5. M. A. Di Gangi, M. Negri, M. Turchi, ASRU, 14-18 Dec. 2019, Singapore (2019)
6. M.Cheng, J.Sheu, N. V. Cuong, Y. C. Kuo, GLOBECOM 2020, 7-11 Dec. 2020, Taipei, Taiwan (2020)
7. J. C. Heck, F. M. Salem, MWSCAS, 6-9 Aug. 2017, Boston, MA, USA (2017)
8. D. Takeuchi, K. Yatabe, Y. Koizumi, Y. Oikawa, N. Harada, ICASSP, 4-8 May 2020, Barcelona, Spain (2020)
9. J. Li, A. Mohamed, G. Zweig, Y. Gong, ASRU, 13-17 Dec. 2015, Scottsdale, AZ, USA (2015)
10. J. Salazar, K. Kirchhoff, Z. Huang, ICASSP, 12-17 May 2019, Brighton, UK (2019)
11. J. Li, R. Zhao, H. Hu, Y. Gong, ASRU, 14-18 Dec. 2019, Singapore (2019)
12. C. C. Chiu, T. N. Sainath, Y.Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, M. Bacchiani, ICASSP, 15-20 April 2018, Calgary, AB, Canada (2018)
13. E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, Z. Zhu, ASRU, 16-20 Dec. 2017, Okinawa, Japan (2017)
14. T. Hori, S. Watanabe, Y. Zhang, W. Chan, Interspeech 2017, 20-24 August 2017, Stockholm, Sweden (2017)
15. A. Graves, A. R. Mohamed, G. Hinton, ICASSP, 26-31 May 2013, Vancouver, BC, Canada (2013)
16. H. Scheidl, S. Fiel, R. Sablatnig, ICFHR, 5-8 Aug. 2018, Niagara Falls, NY, USA (2018)
17. A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, IEEE Transactions on Pattern Analysis and Machine Intelligence, **31**, 855 – 868 (2008)
18. L.Borgholt, J. D. Havtorn, Ž. Agić, A. Søgaard, L. Maaløe, C. Igel, Interspeech 2020, 25-29 October 2020, Shanghai, China (2020)
19. A. Graves, ICML 2012, Sunday July 1, 2012, Edinburgh, Scotland (2012)
20. J. Park, Y. Boo, I. Choi, S. Shin, W. Sung, NeurIPS 2018, Sun Dec 2nd through Sat the 8th, 2018, Montréal, Canada (2018)
21. A. Anand, T. Chakraborty, N. Park, ECIR 5 December 2016, 20-23 March 2016, Padua, Italy (2016)
22. Duyu Tang, Bing Qin, Xiaocheng Feng, Ting Liu, COLING 3 December 2015, Aug 23, 2014 - Aug 29, 2014, Dublin, Ireland (2014)

23. E. M. Taylor, J. Balazs, Y. Matsuo, Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, September 2017, Copenhagen, Denmark (2017)
24. Y. Li, M. C. Chang, S. Lyu, WIFS, 11-13 Dec. 2018, Hong Kong, China (2018)
25. D. Güera, E. J. Delp, AVSS, 27-30 Nov. 2018, Auckland, New Zealand (2018)
26. J. Xiao, Z. Zhou, ICAICA 2020, 27-29 June 2020, Dalian, China (2020)
27. Y. Miao, M. Gowayyed, F. Metze, ASRU 2015, 13-17 Dec. 2015, Scottsdale, AZ, USA (2015)
28. Z. Huang, G. Zweig, M. Levit, B. Dumoulin, B. Oguz, S. Chang, 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, 8-12 Dec. 2013, Olomouc, Czech Republic (2013)
29. C. Victor, J. Brendan, G. Xavier, T. Jordi, C. S. Fu, ICLR 2018, Mon Apr 30th through May 3rd, 2018, Vancouver CANADA (2018)
30. S. Zhang, E. Loweimi, Y. Xu, P. Bell, S. Renals, Interspeech 2019, 15-19 September 2019, Graz, Austria (2019)
31. R.K. Srivastava, K. Greff, J. Schmidhuber, ICML 2015, July 10 and 11, 2015, Lille Grande Palais, France (2015)
32. H. Le, I. Oparin, A. Messaoudi, A. Allauzen, J. Gauvain, F.Yvon, INTERSPEECH 2011, August 27-31. 2011, Florence, Italy (2011)
33. X. Qiang, G. Cheng, Z. Wang, ICETC, 2010.06.22-2010.06.24, Shanghai, China (2010)
34. S. Liu, D. C. Mocanu, Y. Pei, M. Pechenizkiy, 38th International Conference on Machine Learning, 18-24 July 2021, Virtual (2021)
35. Z. Li, C.Ding, S. Wang, W. Wen, Y. Zhuo, C. Liu, Q. Qiu, W. Xu, X. Lin, X. Qian, Y. Wang, HPCA, 16-20 Feb. 2019, Washington, DC, USA (2019)
36. H. Sak, A. Senior, F. Beaufays, INTERSPEECH 2014, 14-18 September 2014, Singapore (2014)
37. J. Hanson, Y. Yang, K. Paliwal, Y. Zhou, Bioinformatics, **33**, 685–692 (2016)
38. M. Ma, J. Gao, L. Feng, J. Stankovic, NeurIPS 2020, Sun Dec 6th through Sat the 12th, Virtual (2020)
39. A. Donzé, O. Maler, FORMATS 2010, September 8-10, Klosterneuburg, Austria (2010)
40. G. E. Hinton, O.Vinyals, J. Dean, NIPS 2014, Mon Dec 8th through Sat the 13th, Palais des Congrès de Montréal, Montréal CANADA (2014)
41. Y. Luo, Z. Chen, T. Yoshioka, ICASSP 2020, May 4-8 2020, Virtual Barcelona (2020)
42. Y. Luo, N. Mesgarani, ICASSP 2018, 5–20 April 2018, Calgary, Alberta, Canada (2018)
43. J. R. Hershey, Z. Chen, J. L. Roux, S. Watanabe, ICASSP 2016, 20 - 25 March 2016, Shanghai, China (2016)
44. Zhang, X.-D. A Matrix Algebra Approach to Artificial Intelligence; Springer, 2020
45. Speak with signs: Active learning platform for Greek Sign Language, English Sign Language, and their translation, Maria Papatsimouli, Lazaros Lazaridis, Konstantinos-Filippos Kollias, Ioannis Skordas, George F. Fragulis, SHS Web Conf. 102 01008 (2021),DOI: 10.1051/shsconf/202110201008

46. Teaching young learners a foreign language via tangible and graphical user interfaces, Heracles Michailidis, Eleni Michailidi, Stavroula Tavoultzidou, George F. Fragulis, SHS Web Conf. 102 01014 (2021), DOI: 10.1051/shsconf/202110201014
47. Lazaridis, L., Papatsimouli, M., Kollias, K. F., Sarigiannidis, P., & Fragulis, G. F. (2021, July). Hitboxes: A Survey About Collision Detection in Video Games. In *International Conference on Human-Computer Interaction* (pp. 314-326). Springer, Cham.
48. Kollias, K.-F.; Syriopoulou-Delli, C.K.; Sarigiannidis, P.; Fragulis, G.F. The Contribution of Machine Learning and Eye-Tracking Technology in Autism Spectrum Disorder Research: A Systematic Review. Electronics 2021, 10, 2982.
49. Kollias, K. F., Syriopoulou-Delli, C. K., Sarigiannidis, P., & Fragulis, G. F. (2021, July). The contribution of Machine Learning and Eye-tracking technology in Autism Spectrum Disorder research: A Review Study. In *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)* (pp. 1-4). IEEE.
50. Fragulis, G. F., Papatsimouli, M., Lazaridis, L., & Skordas, I. A. (2021). An Online Dynamic Examination System (ODES) based on open source software tools. Software Impacts, 7, 100046.
51. Kelli, V., Argyriou, V., Lagkas, T., Fragulis, G., Grigoriou, E., & Sarigiannidis, P. (2021). IDS for industrial applications: a federated learning approach with active personalization. *Sensors*, *21*(20), 6743.
52. Pliatsios, D., Sarigiannidis, P., Fragulis, G., Tsiakalos, A., & Margounakis, D. (2021, June). A Dynamic Recommendation-based Trust Scheme for the Smart Grid. In *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)* (pp. 464-469). IEEE.
53. Radoglou-Grammatikis, P., Sarigiannidis, P., Efstathopoulos, G., Lagkas, T., Fragulis, G., & Sarigiannidis, A. (2021, June). A Self-Learning Approach for Detecting Intrusions in Healthcare Systems. In *ICC 2021-IEEE International Conference on Communications* (pp. 1-6). IEEE.