FULLY CONVOLUTIONAL STRUCTURED LSTM NETWORKS FOR JOINT 4D MEDICAL IMAGE SEGMENTATION

Yang Gao[1], *Jeff M. Phillips*[1], *Yan Zheng*[1], *Renqiang Min*[2], *P. Thomas Fletcher*[1], *Guido Gerig*[3]

[1]School of Computing
University of Utah
Salt Lake City, UT 84112

[2]Machine Learning Department
NEC Laboratories America
Princeton, NJ 08540

[3]Tandon School of Engineering
New York University
Brooklyn, NY 11201

ABSTRACT

Longitudinal medical image analysis has great potential to reveal developmental trajectories and monitor disease progression. This process relies on consistent and robust joint 4D segmentation. Traditional methods highly depend on the similarity of images over time and either build a template or assume the images could be co-registered. This process may fail when image sequences present major appearance changes. Recently, deep learning (DL) approaches have achieved state-of-the-art results for related challenges in computer vision. These approaches make use of models such as fully convolutional networks (FCNs) for end-to-end pixel-wise segmentation and recurrent neural networks (RNNs) with long short-term memory (LSTM) units for sequence-to-sequence modeling. In this paper, we propose a new DL framework called *FCSLSTM* for 4D image segmentation with FCNs for the spatial model and LSTM for the temporal model. This is the first DL framework with deep integration of FCNs and LSTM for joint 4D segmentation that could be trained end-to-end. Our approach achieves promising results with the demonstrated application to longitudinal pediatric magnetic resonance imaging (MRI) segmentation.

1. INTRODUCTION

Longitudinal MR image analysis plays an important role in studying dynamic changes among individual subjects over time. It is paramount towards research in brain development, degeneration and follow-up disease progression, and reliable tissue segmentation is the first step in this analysis. The strong correlation presented in a 3D MRI series for an individual subject over time drives the development of the joint 4D segmentation in order to improve segmentation consistency. A joint 4D segmentation is essential for domains like infant MRIs where the appearance and shape of white and gray matter change dramatically due to tissue myelination. The example in the first row of the Fig. 4 illustrates that it is not at all obvious how to predict the middle image from the other four images – as would be necessary in a model used for longitudinal analysis.

Previous works in the field either require an image template [1, 2] or specifically designed registration [3, 4, 5] to aid segmentation. In addition, these methods typically need a complex data argument, e.g. a special intensity normalization, for each specific dataset. In contrast, machine learning methods may only need very simple data argumentation and a reasonable size of training data. Fully convolutional networks [6] (FCNs) were developed for semantic segmentation of natural images and have rapidly found applications in biomedical image segmentations, such as electron microscopic (EM) images [7] and MRI [8, 9], due to its powerful end-to-end training. The 3D u-net [10] extends the network for volumetric segmentation that learns from sparsely annotated slices by setting the weights of unlabeled voxels in the loss to zero. Recently, convolutional LSTMs were developed to explicitly deal with 2D input. Shi et al. [11] use convolution in the spatial dimension and LSTM in the time dimension for precipitation nowcasting. Chen et al. [12] use LSTM over individual slices in a 3D volume and in each slice use a revised u-net [7] for EM image segmentation to explicitly leverage anisotropic 3D image resolution. However they have not train the model end-to-end in practice.

In this paper, we propose a network integrating FCNs with LSTM for joint 4D segmentation of MRIs. A bi-directional LSTM is used to model the correlation of images over time. At a single time point, a *structured LSTM* is developed as a generalization of convolutional LSTM, which allows us to stack complex LSTMs into a deep network with FCNs' architecture without suffering from efficiency loss. This particular structure allows for a significantly smaller and more efficient network than the state-of-the-art [7] without sacrificing accuracy. Ultimately, we arrive at a concise and thoroughly structured architecture that is highly *accurate*, and by virtue of its small number of parameters, is efficient to train the whole network *end-to-end* from scratch, and we demonstrate feasibility with applications on two clinical brain image datasets.

Acknowledgements: This work is supported by NIH grants NA-MIC Roadmap U54 EB005149, P01 DA022446 and 1R01 DA038215-01A1. We acknowledge the NIH grant ACE RO1 HD 055741 for providing longitudinal infant MRI data, Weili Lin and Dinggang Shen at UNC Chapel Hill for sharing the multi time-point infant MRI series and the support of NVIDIA Corporation with the donation of GPUs used for this research.

2. METHOD

2.1. A Concise Fully Convolutional Network

Convolutional Neural Networks (CNNs) have recently produced excellent performance in image classification. Although many variants of CNNs have been developed, basic components of CNNs stay the same; i.e. they have four types of layers: the convolutional, non-linearity, pooling and fully-connected layers. The same holds for fully convolutional networks (FCNs). Various FCNs have been used for pixel-level image segmentation where besides those four layers, they employ two additional layer types. One is usually called the deconvolutional layer used for up-sampling the smaller feature maps back to the origin image size. The other is called crop layer, which is used to crop the feature maps to a desired dimension. Typically the design of FCNs starts from a traditional CNNs and then transfers the fully-connected layers to the convolutional layers with kernels of size 1×1 (we see the above layers as the front-end, i.e. the down-sampling part), and then add deconvolutional, convolutional, and crop layers (we see these layers as the back-end, i.e. up-sampling part). Within the back-end, the authors in [6] discuss using feature maps from the front-end convolutional layers to feed back into the back-end. We refer this design choice as the *fuse level*. A larger fuse level indicates that more feature maps originating from different convolutional layers in the front-end are used. Because VGGs [13] networks are very regular in design (i.e. in each VGG layer, the convolutional layer does not change the dimension of feature map, only the pooling layer would do), they are commonly chosen as the base architecture for FCNs. One known concern about VGGs is the huge number of the modeling parameters which lead to a huge memory consumption. This would be even worse if we add more layers to FCNs. To solve this problem, starting with a standard VGG-16-layers as the front-end, we make extended empirical tests with different FCN architectures on volumetric MRIs datasets of which typical dimension is 256^3. We have some observations from these empirical tests about the back-end: 1) Without sacrificing accuracy, the number of convolutions can be fixed as 1 and the number of output channels in the convolutional layers could be fixed as the number of classes in the segmentation; 2) Instead of transferring the fully-connected layers to 1×1 convolutional layers, the fully-connected layers could be avoided; 3) High fuse level leads to more detailed and refined segmentation results. Besides the back-end, in the front-end, the number of input and output channels in each convolutional layer is another factor to impact the number of parameters. We test shrinking the number of channels used in VGG by factors of 2 and 4 and we find a slight accuracy decrease while resulting in a huge reduction of model's parameters, which is probably acceptable. Ultimately, with all these considerations, we propose a more concise yet powerful FCN architecture shown in Fig. 1 and Table 1, which is 20 time smaller than u-net [7] and 130 times smaller than the original FCNs proposed in [6].

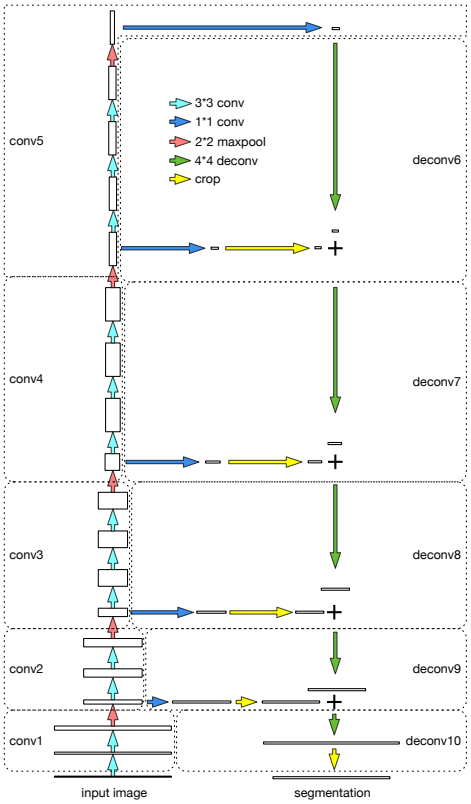

Fig. 1. Our proposed FCN architecture. Each small rectangle corresponds to a feature map. The arrows denote the different operations. Dashed boxes with the names represent layers in the FCN.

Table 1. The parameters in each layers of our concise FCN. The nInput and nOutput show the number of input and output channels in the convolutional layer. The n represents the channel number of input image, and the c represents the number of classes in prediction.

Network Layers		The parameters of the convolutional layers				
layers	sub-layers	nInput	nOutput	kernel	stride	pad
conv1	conv1_1 + relu1_1	n	16	3	1	100
	conv1_2 + relu1_2 + pool1	16	16	3	1	1
conv2	conv2_1 + relu2_1	16	32	3	1	1
	conv2_2 + relu2_2 + pool2	32	32	3	1	1
conv3	conv3_1 + relu3_1	32	64	3	1	1
	conv3_2 + relu3_2	64	64	3	1	1
	conv3_3 + relu3_3 + pool3	64	64	3	1	1
conv4	conv4_1 + relu4_1	64	128	3	1	1
	conv4_2 + relu4_2	128	128	3	1	1
	conv4_3 + relu4_3 + pool4	128	128	3	1	1
conv5	conv5_1 + relu5_1	128	128	3	1	1
	conv5_2 + relu5_2	128	128	3	1	1
	conv5_3 + relu5_3 + pool5	128	128	3	1	1
	conv5_4	128	c	1	1	0
deconv6	deconv6_1	c	c	4	2	1
	conv6_1 + crop6 + sum6	128	c	1	1	0
deconv7	deconv7_1	c	c	4	2	1
	conv7_1 + crop7 + sum7	64	c	1	1	0
deconv8	deconv8_1	c	c	4	2	1
	conv8_1 + crop8 + sum8	32	c	1	1	0
deconv9	deconv9_1	c	c	4	2	1
	conv9_1 + crop9 + sum9	16	c	1	1	0
deconv10	deconv10_1 + crop10	c	c	4	2	0
total number of parameters		9.2×10^5				

2.2. The structured LSTM

The recurrent neural networks (RNNs) have the capacity of repeatedly learning based on previously remembered information, and then storing this learned new information. The RNNs are networks with loops of recurrent units in them, allowing information to persist. The long short-term memory [14] (LSTM) is one of the recurrent units invented to ease the "gradients vanishing" issue [15]. Recently, the convolutional LSTM was developed [16, 17, 11] to explicitly deal with spatial 2D input. In this paper, we generalize the LSTM further and consider any operators that could be applied to the input of LSTM. Given the input data x_t at current time point t, the hidden state input h_{t-1} and the cell state input c_{t-1} at previous time point $t-1$, the structured LSTM update formula is shown in Eq (1) and Fig. 2,

$$
\begin{aligned}
i_t &= \sigma(\Psi_{xi}(x_t) + \Psi_{hi}(h_{t-1}) + b_i) \\
f_t &= \sigma(\Psi_{xf}(x_t) + \Psi_{hf}(h_{t-1}) + b_f) \\
o_t &= \sigma(\Psi_{xo}(x_t) + \Psi_{ho}(h_{t-1}) + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(\Psi_{xc}(x_t) + \Psi_{hc}(h_{t-1}) + b_c) \\
h_t &= o_t \circ \tanh(c_t)
\end{aligned}
\tag{1}
$$

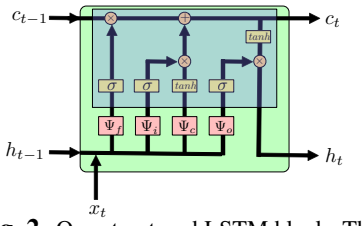

Fig. 2. One structured LSTM block. The green, red and purple boxes represent the structured LSTM block, the arbitrary operator and the update path respectively.

where i_t, f_t and o_t are input, forget, output gates, c_t and h_t are the new cell and hidden states at time t, $\sigma(x) = \frac{1}{1+e^{-x}}$, $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, and $\circ$ are element-wise matrix multiplication (i.e. Hadamard product). Notice that Ψ denotes any operators and the operators could be different for the input data x and the hidden state h. For example, if Ψ is the linear operator, the update Eq (1) becomes the classic LSTM [14]; if Ψ is the convolution operator, the update Eq. (1) represents the convolutional LSTM [11]. There are some advantages with this generalization including: 1) This adds flexibility to use complex operators, for example a differential operator if meaningful, on the input of LSTM, which made LSTM can handle any dimension of data, or even other complex data structure; 2) Combination of a few functions as an operator Ψ makes LSTM only accumulate the useful and meaningful information over time. One application in VGG could be that we could apply all conv1_1+relu1_1+conv1_2+relu1_2 as one operator Ψ. Therefore, only the features from conv1 of VGG save into LSTM memory rather than the features from every internal convolutional sub-layer, such as conv1_1 and conv1_2. This could save a lot of memory during training since training by back propagation through time (BPTT) [18, 19] needs to keep every status of the LSTM block over time in memory. 3) If we stack LSTM layers together to build a deep stacked LSTM network, we could use any meaningful and different operators in each LSTM layers. We name this new generalization *the structured LSTM* because of the capacity of building a structured network inside LSTM block or through stacking layers.

2.3. Fully Convolutional Structured LSTM Networks

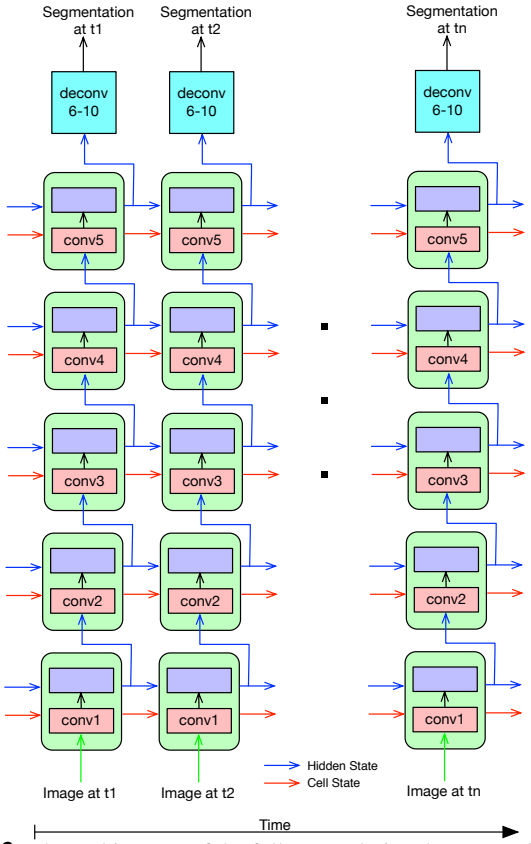

Fig. 3. The architecture of the fully convolutional structured LSTM network. Notice the color scheme is similar to Fig. 2.

The fully convolutional structured LSTM networks (FCSLSTM) is the combination of FCN and LSTM, and is a stacked structured LSTM. Based on our test, only the features from the 5 convolutional layers in the front-end may take part in the segmentation. Therefore, the architecture of the FCSLSTM contains 5 layers stacked structured LSTM followed by all back-end layers from deconv6 to deconv10 and finally a softmax layer to generate the segmentation probability map. The conv1, which includes conv1_1+relu1_1+conv1_2+relu1_2, is used as the operator Ψ of the input x in the first stacked structured LSTM layer, the conv2 is used as the operator Ψ of the input x in the second stacked structured LSTM layer, and etc. In all LSTM layers, the convolution operator is used for the hidden state h. The Fig. 3 shows the FCSLSTM in details. The weighted cross-entropy loss is used, and the weights are inverted proportionally to the number of pixels in each class in the training dataset. In the case of longitudinal MRI data, both past and future contexts are helpful to improve current estimates. So we also use bidirectional RNNs (BRNNs) [20]. The basic

idea of BRNNs is to present each training sequence forwards and backwards to two separate recurrent hidden layers with the same inputs, both of which are connected to the same output layer. For training, the forward pass for the BRNN hidden layers is the same as for a RNN, except that the input sequence is presented in opposite directions to the two hidden layers and the output layer is not updated until both hidden layers have processed the entire input sequence. Similarly, the backward pass proceeds as for a standard RNN trained with BPTT [18, 19], except that all the output layer gradient terms are calculated first, then fed back to the two hidden layers in opposite directions.

3. EXPERIMENTS

To evaluate our model, we use 2 datasets: 1) The BRIC clinical dataset (UNC Chapel Hill Biomedical Research Imaging Center), which contains multimodal (T1w and T2w) pediatric longitudinal MRI scans of 10 subjects at 5 time points within the first year after birth. Only 1 slice per 3D volume is manually labeled by an expert, so that there are only 50 labeled 2D images. 2) The Autism Center of Excellence IBIS clinical study of subjects at high-risk for autism, which also includes multimodal pediatric longitudinal MRI scans at 3 time points within the firtst two years of life. We only use a sample (10 subjects) from the large IBIS dataset. There is no manual expert labeling, but the MRIs of the 2nd and 3rd could be segmented via conventional template-moderated EM segmentation to act as pseudo ground truth for training our model. Our code is written in torch and running on NVIDIA Titan X video card with 12G vram, except the bi-directional LSTM model which does not fit into the GPU memory. We use Kaiming initialization [21] and RMSprop optimization with smooth constant $\alpha = 0.95$, initial learning rate 0.001. The learning rate gradually decays after 20 epochs with factor 0.97. For quantitative evaluation, the mean pixel accuracy (mPA) metric is used, which computes a ratio between the amount of properly classified pixels and the total number of them in a per-class basis and then averaged over the total number of classes.

The BRIC dataset is used to compare different models. The dataset is split to use the 1st, 2nd, 4th and 5th time points as training sets and the 3rd time points as test set. From Fig. 4, we observe that FCSLSTM improves the segmentation results over FCN only, and the bi-directional FCSLSTM achieves the best result. One interesting observation is that by using LSTM to find correlations over time, the segmentations of the training sets are also improved. Ultimately, the mPA of FCN only, FCSLSTM and bi-directional FCSLSTM achieve *92%, 93.5% and 94.8%* for training sets and 85%, 85.3% and 86.3% for the test set. These results are promising considering very limited training data (only 40 2D images).

The IBIS dataset contains sufficient training data. We use the 1st time point as test set, and the 2nd and 3rd time point as training sets. Although lack of ground truth, visual inspection of Fig. 5 indicates a good segmentation result visually. Please notice that for all 3 datasets, we choose the most challenging time points as test sets (MRI between 3 and 9 months for pediatric studies, e.g.).

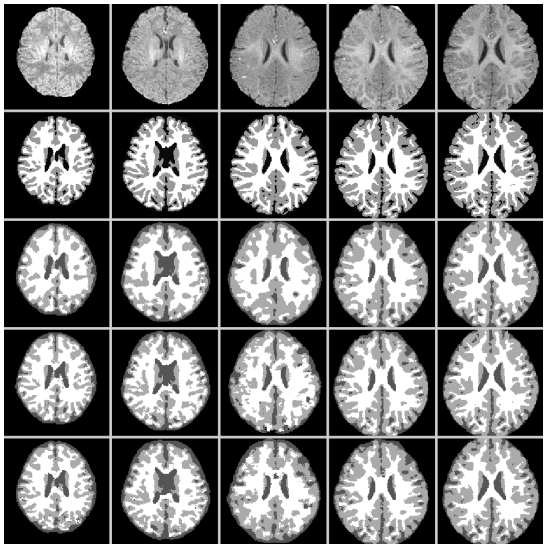

Fig. 4. The BRIC dataset with very limited training data. The columns from left to right are at the 0, 3rd, 6th, 9th and 11th month. The rows from top to bottom show the input, the ground truth, the segmentation by FCN only, the segmentation by FCSLSTM, and the segmentation by bi-directional FCSLSTM.

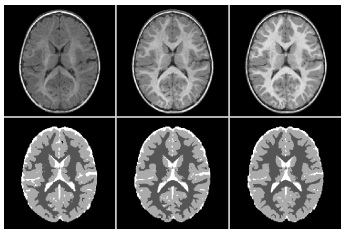

Fig. 5. The IBIS dataset. The columns from left to right are at the 6th, 12th and 24th month. The rows from top to bottom show the input and segmentation by FCSLSTM.

4. CONCLUSION

We have presented a novel DL approach for joint 4D segmentation of longitudinal MRI. Unlike temporal image series showing similar contrast appearance over time, we additionally approach the challenging issue of considerable appearance changes due to brain maturation as observed in early infant brain imaging. To our knowledge, this is the first proposal of deep integration of FCNs and a gneralized *structured* LSTMs, called *FCSLSTM*, for spatial and temporal modeling of image shape and appearance changes, to be trained end-to-end. A concise FCN is introduced for efficient end-to-end training on GPUs with constrained memory. We obtain promising results on real clinic datasets, with the limitation that only incomplete expert segmentations of brain tissue are available for training and validation. Future work will focus on additional testing of the methodology itself but also extended validation of tissue segmentation results based on newly available expert labeling.

5. REFERENCES

[1] M. Reuter, N. J. Schmansky, H. D. Rosas, and B. Fischl, "Within-subject template estimation for unbiased longitudinal image analysis," *NeuroImage*, vol. 61, no. 4, pp. 1402 – 1418, Mar. 2012.

[2] F. Shi, Y. Fan, S. Tang, D. Shen, and et al., "Neonatal brain image segmentation in longitudinal MRI studies," *NeuroImage*, vol. 49, no. 1, pp. 391–400, Jan. 2010.

[3] Zhong Xue, Dinggang Shen, and Christos Davatzikos, "Classic: consistent longitudinal alignment and segmentation for serial image computing," *NeuroImage*, vol. 30, no. 2, pp. 388–399, 2006.

[4] Marc Niethammer, Gabriel L Hart, Danielle F Pace, Paul M Vespa, Andrei Irimia, John D Van Horn, and Stephen R Aylward, "Geometric metamorphosis.," *Audio and Electroacoustics Newsletter, IEEE*, vol. 14, no. Pt 2, pp. 639–646, Jan. 2011.

[5] Yang Gao, Miaomiao Zhang, Karen Grewen, P Thomas Fletcher, and Guido Gerig, "Image registration and segmentation in longitudinal mri using temporal appearance modeling," in *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*. IEEE, 2016, pp. 629–632.

[6] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[8] Dong Nie, Li Wang, Yaozong Gao, and Dinggang Sken, "Fully convolutional networks for multi-modality isointense infant brain image segmentation," in *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*. IEEE, 2016, pp. 1342–1345.

[9] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, 2016, pp. 565–571.

[10] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2016, pp. 424–432.

[11] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, 2015, pp. 802–810.

[12] Jianxu Chen, Lin Yang, Yizhe Zhang, Mark Alber, and Danny Z Chen, "Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation," in *Advances in Neural Information Processing Systems*, 2016, pp. 3036–3044.

[13] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

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

[15] Yoshua Bengio, Patrice Simard, and Paolo Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 157–166, 1994.

[16] Marijn F Stollenga, Wonmin Byeon, Marcus Liwicki, and Juergen Schmidhuber, "Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation," in *Advances in Neural Information Processing Systems*, 2015, pp. 2998–3006.

[17] Viorica Patraucean, Ankur Handa, and Roberto Cipolla, "Spatio-temporal video autoencoder with differentiable memory," *arXiv preprint arXiv:1511.06309*, 2015.

[18] Ronald J Williams and David Zipser, "Gradient-based learning algorithms for recurrent networks and their computational complexity," *Back-propagation: Theory, architectures and applications*, pp. 433–486, 1995.

[19] Paul J Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[20] Mike Schuster and Kuldip K Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.