4D CONVOLUTIONAL NEURAL NETWORKS FOR MULTI-SPECTRAL AND MULTI-TEMPORAL REMOTE SENSING DATA CLASSIFICATION

Michalis Giannopoulos[⋆†] *Grigorios Tsagkatakis*[⋆†] *Panagiotis Tsakalides*[⋆†]

[⋆] Department of Computer Science, University of Crete, Heraklion, 70013, Greece
[†] Foundation for Research and Technology Hellas (FORTH), Heraklion, 70013, Greece

ABSTRACT

Multi-temporal remotely sensed observations acquired by multi-spectral sensors contain a wealth of information related to the Earth's state. Deep learning methods have demonstrated a great potential in analyzing such observations. Traditional 2D and 3D approaches are unable to effectively extract valuable information encoded across all available dimensions. In this work, we propose the extension of current fully-convolutional models for multi-temporal remote sensing data classification to their high-dimensional analogs, which can naturally capture multi-dimensional dependencies and correlations. Experimental analysis on recently released observations from Landsat-8 reveals that our proposed high-dimensional fully-convolutional approach exhibits a clear classification performance improvement over state of the art low-order fully-convolutional models.

Index Terms— Convolutional Neural Networks, Time-series, Remote Sensing, Land-Cover Classification.

1. INTRODUCTION

Multi-Spectral (MS) and *Multi-Temporal* (MT) *Remote Sensing* (RS) data hold a great promise for analysis tasks including land-cover classification. Unlike conventional *Machine Learning* (ML) methods [1–3], *Deep Learning* (DL) architectures such as *Convolutional Neural Networks* (CNNs), can be trained through an end-to-end learning process, allowing them to perform impressively in tasks including image enhancement [4], scene [5] and land-cover [6] classification.

However, traditional approaches employ 2D-CNNs which are not directly applicable to the MT-RS classification task, requiring the temporal information to be collapsed. To overcome this obstacle, Kussul *et al.* proposed in [7] a technique of tying together a 2D-spatial CNN with a 1D-spectral CNN, whereas in [8] a 3D-CNN fully exploits spatial and temporal feature learning simultaneously. Methods based on the so-called *Temporal-CNNs* [9–11], operating on (temporal) *Normalized Difference Vegetation Index* (NDVI) representations rather than on raw MT-RS data, have also been proved promising. Other types of DL-based approaches for tackling the MT-RS land-cover classification problem include *RNN* [12] and *LSTM* [13] networks. These methods are by-design suitable for time-series modelling, however, they do not inherently encode spatial context.

In this work, we propose a 4D-CNN approach to the MT-RS land-cover classification task, for effectively learning spatio-spectro-temporal features at the same time. Extensive experiments on a recently released MT-RS dataset demonstrate the potential of the proposed higher-order data encoding approach across all examined scenarios. All in all, the key contributions of this paper can be summarized as follows: (i) Introduction of 4D-CNNs to tackle the MT-RS land-cover classification task; (ii) Effective exploitation of high-order data correlations without any loss of information; (iii) Demonstration of the 4D-CNN superiority over their lower-dimensional counterparts and state-of-the-art methods.

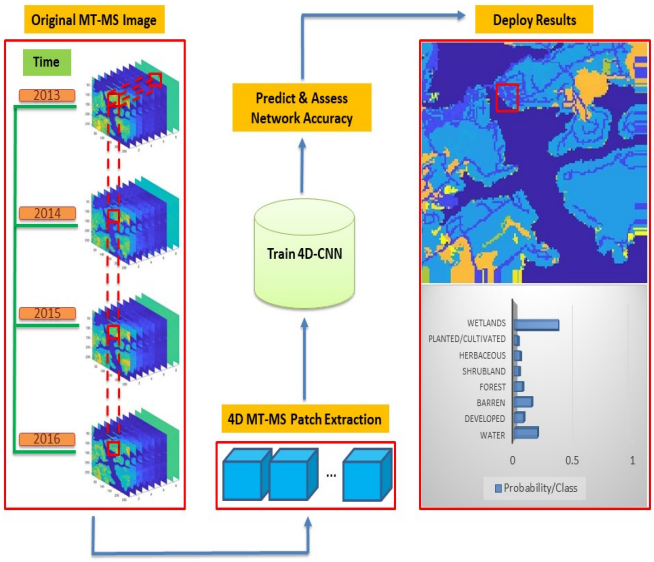


Fig. 1: The proposed method pipeline for the MT-RS land-cover classification task. Spatial patches are extracted across all available spectral bands and time instances of the MT-RS imagery, and subsequently fed to the proposed 4D-CNN system charged with the task of pixel-level classification.

The authors would like to thank the IEEE GRSS Image Analysis and Data Fusion Technical Committee and Microsoft for organizing the Data Fusion Contest.

2. PROPOSED METHOD

To jointly exploit the spatial, spectral and temporal information of the MT-RS data, we introduce a 4D convolutional layer. With this configuration, the value of a convolved output neuron at position (k, l, m, n) can be expressed as follows:

$$\begin{aligned} y_{k,l,m,n} &= f\Bigg(\sum_{c}^{C_{in}} \sum_{s=0}^{S-1} \sum_{t=0}^{T-1} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \\ &\qquad w_{i,j,t,s} x_{c,(k+i)(l+j)(m+t)(n+s)} + b_{i,j,t,s} \Bigg) \\ &= f\Bigg(\sum_{s=0}^{S-1} \Bigg[\sum_{c}^{C_{in}} \sum_{t=0}^{T-1} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \\ &\qquad w_{i,j,t,s} x_{c,(k+i)(l+j)(m+t)(n+s)} \Bigg] + b_{i,j,t,s} \Bigg) \\ &= f\Bigg(\sum_{s=0}^{S-1} C_{3D} + b_{i,j,t,s} \Bigg) \end{aligned} \tag{1}$$

where $f(.)$ is an activation function, $w_{i,j,t,s}$ stands for the value of the kernel connected to the current feature map at the position (i, j, t, s), $x_{c,(k+i)(l+j)(m+t)(n+s)}$ represents the value of the input neuron at input channel c, $b_{i,j,t,s}$ is the bias of the computed feature map, C_{in} denotes the number of original channels (i.e. first layer) or the number of feature maps of the previous layer (i.e. intermediate layer), C_{3D} stands for the 3D-convolution operator, and H, W, T and S are the height, width, temporal length and spectral bands, respectively.

The re-arrangement of the convolution sums shown in (1) is feasible since convolution is a linear operation and therefore the order of summation can be changed. Such a transformation was proposed in [14, 15], and ends up with stacking multiple sequences of 3D convolutions -the term C_{3D} in (1)- along the fourth dimension. From an implementation point of view, further re-arrangement of the respective for-loop took place as in [14] (i.e. 3D input-frames convolved with respective 3D filter-frames) in order for our custom layer to implement true (instead of separable) 4D convolution.

As illustrated in Fig. 1, in order to efficiently train our 4D model with several samples, we extract overlapping patches around each pixel of the raw MT-RS imagery. Those sample patches are used to train the 4D-CNN, which in turn predicts the label of the center-pixel of equivalent test samples. The rationale for the introduction of our 4D-CNN model architecture is two-fold: first, since the training samples to be used are pixel-centered patches, we do not take into consideration any scaling or translation factors (i.e. no need for pooling layers); second, works on *Hyper-Spectral* pixel-level classification [16, 17] have shown that in such experimental scenarios fully-convolutional neural networks are excellent feature learners.

Concerning the topology of the designed model, all convolutional layers are used with a small kernel size equal to 3 across every dimension, with same padding and unit strides across all dimensions in order to preserve as much information as possible across every dimension before the final fully-connected (prediction) layer, containing the number of different classes. Each convolutional layer is directly followed by a batch-normalization layer and a ReLU-activation layer. The same stack-of-layers (i.e. Conv-BN-ReLU) is repeated up to 4 times, in order to find the best architecture of the CNN model based on the validation set results. Depending on how many convolutional layers are used, $32 - 32 - 64 - 64$ filters are employed per each layer accordingly, while a batch-size of 128 samples is used for every designed model. Regarding the optimization process of the network, we employed the Adam scheme [18] with a constant learning rate of 0.0001 and exponential decay rates for the 1^{st} and 2^{nd} moment estimates equal to $\beta_1 = 0.9$ and $\beta_2 = 0.999$, respectively. Categorical cross-entropy was used as the loss function.

3. EXPERIMENTAL EVALUATION

3.1. Experimental Setup

For validating experimentally our proposed approach, we considered a large MT-RS dataset which was released recently in the context of the IEEE GRSS Data Fusion Contest[1] for the problem at hand, providing 2250 different tiles (each one covering approximately a 4km $\times$ 4km area) over the state of Maryland, USA. For each tile, they are provided Landsat-8 training data (i.e. 9-band MS satellite data, at 30m spatial resolution collected once a year through $2013 - 2017$) as well as *USGS National Land Cover Database* ground-truth labels for years 2013 and 2016 (i.e. 15-class land-cover data, at 30m spatial resolution). In the present study, ground-truth labels are considered those of 2016, while training data are considered the respective MT-RS image tiles through $2013 - 2016$. For the upcoming experiments, we randomly selected an image tile (namely, tile$_{3999}$) whose respective labels do not contain any pixels with undefined values (i.e. pixels with values equal to 0), since these pixels do not form any meaningful class out of the 15 available ones. Subsequently, we adopted the following pre-processing steps for the selected tile: (i) spatial padding until its original spatial size reaches 4096×4096; (ii) spatial subsampling every 16 pixels across height and width; (iii) spatial patch-extraction across all spectral bands and all time-instances. These patches are overlapping, extracted for all possible locations in the data, and labeled using the center-pixel of each patch. In this way, the patch-extraction process serves as a data augmentation technique, since each selected tile generates $256 \times 256 = 65536$ training samples of size $(p, p, 9, 4)$ each.

[1] "[REF. NO.] 2021 IEEE GRSS Data Fusion Contest. Online: https://www.grss-ieee.org/community/technical-committees/2021-ieee-grss-data-fusion-contest-track-msd/"

With this setup, we split in a random way the aforementioned available samples into $60\%/20\%/20\%$ non-overlapping training/validation/test sets, and train the proposed 4D-CNN model using a constant number of up to 100 epochs. We select and report the model's performance based on the best validation accuracy it achieved during the training process (irrespective if that was on the final epoch or not). This best performing model is then used in the test set, once, in order to assess the final performance of the method. All designed architectures were developed in Python programming platforms, by exploiting the open-source ML framework TensorFlow [19] and the higher-level DL-specific library Keras [20], due to the desired and provided high level of customization of a DL model as well as fast GPU-performed calculations. In our experiments, we used NVIDIA's GPU model *Quadro P4000*, with 8Gb available RAM memory.

3.2. 4D-CNN Architectures Parameter Tuning

To understand the impact of input data dimensionality, we initially construct a 2D-CNN model, in which only spatial features are learned whereas the spectral and temporal dimensions are averaged (i.e. 2D-CNN). Moreover, we construct two different 3D-CNN models, one for learning spatio-temporal features (i.e. 3D-CNN-T: averaging the spectral dimension) and one learning spatio-spectral ones (i.e. 3D-CNN-S: averaging the temporal dimension).

Table 1: Classification accuracy and respective training time (minutes) for the trained CNN models.

Model	Patch-Size	#Stack-of-Layers	Accuracy	Time
2D-CNN	$(5,5,1,1)$	3	0.5677	4.238
	$(5,5,1,1)$	4	0.5720	4.732
	$(7,7,1,1)$	3	0.5986	4.231
	$(7,7,1,1)$	4	**0.6049**	**4.928**
3D-CNN-T	$(5,5,1,4)$	3	0.7143	6.122
	$(5,5,1,4)$	4	0.7527	9.151
	$(7,7,1,4)$	3	0.7699	8.958
	$(7,7,1,4)$	4	**0.8080**	**14.558**
3D-CNN-S	$(5,5,9,1)$	3	0.7344	9.792
	$(5,5,9,1)$	4	0.7463	16.124
	$(7,7,9,1)$	3	0.7682	16.121
	$(7,7,9,1)$	4	**0.7953**	**27.435**
4D-CNN	$(5,5,9,1)$	3	0.8767	60.489
	$(5,5,9,1)$	4	0.8813	111.836
	$(7,7,9,1)$	3	**0.8916**	**107.619**
	$(7,7,9,1)$	4	0.8906	204.110

In a first set of experiments, our objective is to assess the performance of the employed 4D-CNN and its lower-dimensional counterparts relative to the number of stack-of-layers they consist of. Then, we quantify the impact of the patch-size the CNN models are trained with by obtaining more spatial information across each centered pixel, while the spectral and temporal ones remain set. As it can be seen in Table 1, as we employ more layers the obtained accuracy improves for all models, indicating that the networks' generalization capability is boosted. In addition, spatially-richer input patches increase the learning capacity of the networks, irrespective of the number of layers employed. Unavoidably, more layers means that more parameters must be learned, which is depicted in the time needed to train the networks. All in all, we can conclude that the proposed 4D-CNN model achieves an accuracy improvement of up to 8.36% over the second best model at the cost of the tolerable ~ 2 hours time needed for training.

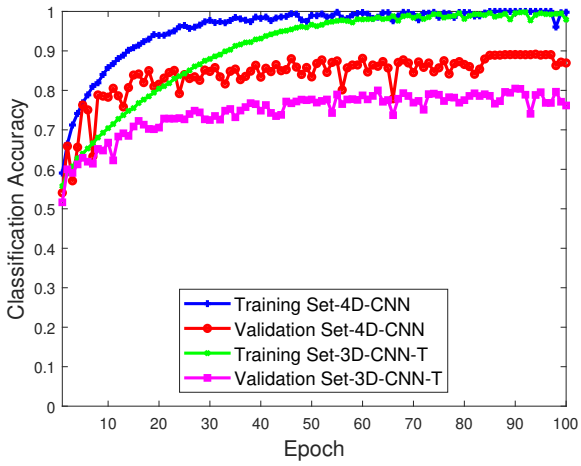
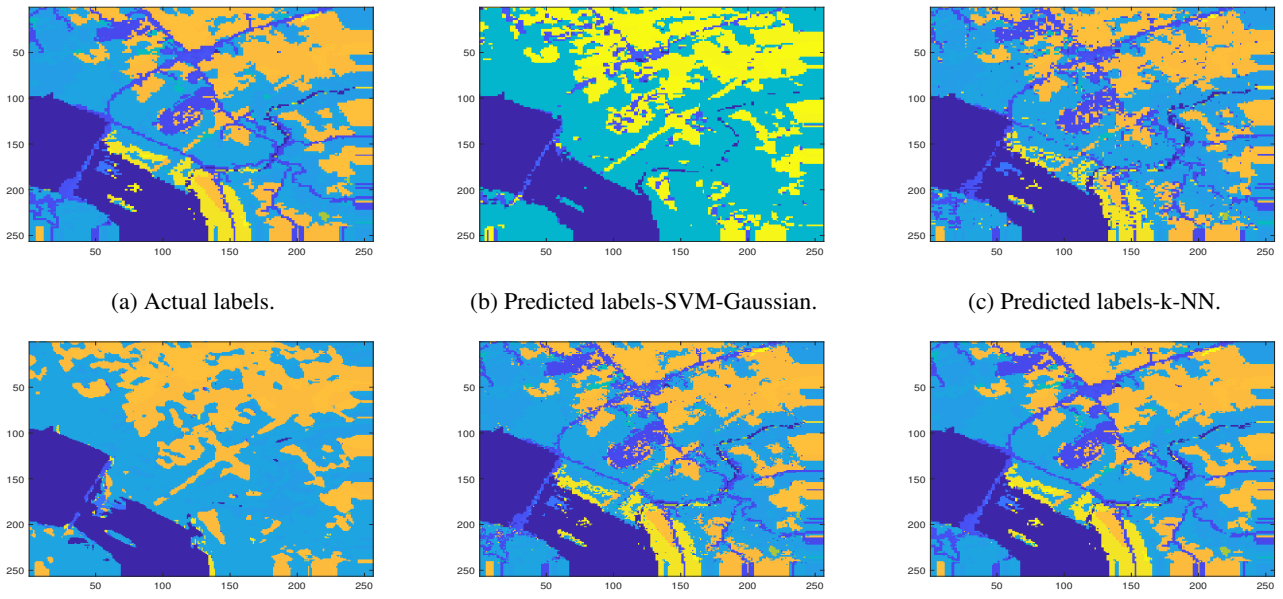


Fig. 2: Classification accuracy during the training process of the best 4D and 3D-CNN models. As the number of epochs increases, so does the performance of the CNN models, with a clear improvement in favour of the proposed 4D scheme.

Figure 2 illustrates the training process in terms of classification accuracy for the training and validation sets using the best hyper-parameters. We observe that the performance of the 4D-CNN model improves gradually, while at the same time over-fitting issues are prevented as the validation curve tracks fairly well the training curve, indicating a strong generalization capability. On the other hand, the best 3D-CNN-T model converges slower and to a lower classification accuracy value, when compared to the proposed 4D model.

3.3. Comparison to State-of-the-Art and ML Methods

We subsequently consider the best CNN models of each dimensionality as they arise from Table 1, and compare them with the state-of-the-art CNN architecture of [8], namely a 3D network consisting of convolutional, average-pooling and fully-connected layers. In addition, we train a Gaussian SVM, and a k-NN (with $k = 5$) classifier, with collapsed input patches, in order to compare with some non-DL based methods, as a baseline.



(a) Actual labels. (b) Predicted labels-SVM-Gaussian. (c) Predicted labels-k-NN.

(d) Predicted labels-3D-CNN [8]. (e) Predicted labels-3D-CNN-T. (f) Predicted labels-4D-CNN (proposed).

Fig. 3: Actual and predicted labels corresponding to tile 3999, obtained via the the proposed 4D-CNN and its competing models. The proposed higher-dimensional model captures better the texture of the map, as well as more of its actual labels/values.

Table 2: Classification accuracy, training time (minutes) and F_1-Score of the proposed method and competing ones.

Model	Accuracy	Time	F1-Score
k-NN	0.7443	0.000	0.5747
SVM-Gaussian	0.5945	2.006	0.4995
3D-CNN [8]	0.7097	6.003	0.5090
2D-CNN	0.6049	4.928	0.3624
3D-CNN-T	0.8080	14.558	0.6509
3D-CNN-S	0.7953	27.435	0.6552
4D-CNN-Proposed	**0.8916**	**107.619**	**0.7796**

Table 2 reports the classification accuracy obtained by the various methods in the test set, accompanied by the respective time needed for training the models, and the achieved macro-averaged F_1-score values. Table 2 clearly shows the significant improvement achieved by the proposed 4D-CNN method both in classification accuracy and F_1-score, as compared to all the other methods. The F_1-score obtained by 4D-CNN indicates the robustness of the proposed higher-order model in class-imbalance situations. Of course, improved performance comes at a cost: training a 4D-CNN takes approximately 7 times longer than the second competing method, 3D-CNN-T.

To have a better visual sense of the proposed model's performance, Figure 3 depicts the actual labels of the employed data tile$_{3999}$, accompanied by the predictions obtained from 5 of the methods listed in Table 2. From the classification maps shown in Figure 3, we notice that the lower-dimensional CNN models as well as the ML-based suffer from severe noise-effects, a result of the poor performance to predict correctly several pixels of the data tile. On the contrary, the proposed 4D-CNN model captures best most of the labels of the actual classification map, ending up with clear segmented regions and bearing a strong resemblance to the ground-truth labels.

4. CONCLUSION

In this work, we proposed a 4D fully-convolutional architecture for the problem of classifying MT-RS data. The approach is based on generalizing existing lower-order fully-convolutional models to their high-dimensional analogues, by customizing and modifying the core functionality of convolution. Based on our experimental findings on a recently released MT-RS dataset, we demonstrated that a clearly improved classification accuracy, with respect to state-of-the-art and empirical methods, is feasible if feature learning is performed in the nominal domain of the data.

5. ACKNOWLEDGMENTS

This work was funded by the the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Innovation (GSRI), under the HFRI Faculty Grant 1725 (V4-ICARUS), and by the CALCHAS project (contract no.842560) of the H2020 Framework Program of the European Commission.

6. REFERENCES

[1] Fabian Löw, U Michel, Stefan Dech, and Christopher Conrad, "Impact of feature selection on the accuracy and spatial uncertainty of per-field crop classification using support vector machines," *ISPRS journal of photogrammetry and remote sensing*, vol. 85, pp. 102–119, 2013.

[2] Enrico Blanzieri and Farid Melgani, "Nearest neighbor classification of remote sensing images with the maximal margin principle," *IEEE Transactions on geoscience and remote sensing*, vol. 46, no. 6, pp. 1804–1811, 2008.

[3] Vincent Simonneaux, Benoît Duchemin, D Helson, S Er-Raki, Albert Olioso, and AG Chehbouni, "The use of high-resolution image time series for crop classification and evapotranspiration estimate over an irrigated area in central morocco," *International Journal of Remote Sensing*, vol. 29, no. 1, pp. 95–116, 2008.

[4] Grigorios Tsagkatakis, Anastasia Aidini, Konstantina Fotiadou, Michalis Giannopoulos, Anastasia Pentari, and Panagiotis Tsakalides, "Survey of deep-learning approaches for remote sensing observation enhancement," *Sensors*, vol. 19, no. 18, pp. 3929, 2019.

[5] Michalis Giannopoulos, Anastasia Aidini, Anastasia Pentari, Konstantina Fotiadou, and Panagiotis Tsakalides, "Classification of compressed remote sensing multispectral images via convolutional neural networks," *Journal of Imaging*, vol. 6, no. 4, pp. 24, 2020.

[6] Marco Castelluccio, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," *arXiv preprint arXiv:1508.00092*, 2015.

[7] Nataliia Kussul, Mykola Lavreniuk, Sergii Skakun, and Andrii Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 778–782, 2017.

[8] Shunping Ji, Chi Zhang, Anjian Xu, Yun Shi, and Yulin Duan, "3d convolutional neural networks for crop classification with multi-temporal remote sensing images," *Remote Sensing*, vol. 10, no. 1, pp. 75, 2018.

[9] Charlotte Pelletier, Geoffrey I Webb, and François Petitjean, "Temporal convolutional neural network for the classification of satellite image time series," *Remote Sensing*, vol. 11, no. 5, pp. 523, 2019.

[10] Nicola Di Mauro, Antonio Vergari, Teresa Maria Altomare Basile, Fabrizio G Ventola, and Floriana Esposito, "End-to-end learning of deep spatio-temporal representations for satellite image time series classification.," in *DC@ PKDD/ECML*, 2017.

[11] Liheng Zhong, Lina Hu, and Hang Zhou, "Deep learning based multi-temporal crop classification," *Remote sensing of environment*, vol. 221, pp. 430–443, 2019.

[12] Marc Rußwurm and Marco Körner, "Multi-temporal land cover classification with sequential recurrent encoders," *ISPRS International Journal of Geo-Information*, vol. 7, no. 4, pp. 129, 2018.

[13] Marc Rußwurm and Marco Körner, "Convolutional lstms for cloud-robust segmentation of remote sensing imagery," *arXiv preprint arXiv:1811.02471*, 2018.

[14] Andriy Myronenko, Dong Yang, Varun Buch, Daguang Xu, Alvin Ihsani, Sean Doyle, Mark Michalski, Neil Tenenholtz, and Holger Roth, "4d cnn for semantic segmentation of cardiac volumetric sequences," in *International Workshop on Statistical Atlases and Computational Models of the Heart*. Springer, 2019, pp. 72–80.

[15] Shiwen Zhang, Sheng Guo, Weilin Huang, Matthew R Scott, and Limin Wang, "V4d: 4d convolutional neural networks for video-level representation learning," *arXiv preprint arXiv:2002.07442*, 2020.

[16] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 4959–4962.

[17] Michalis Giannopoulos, Grigorios Tsagkatakis, and Panagiotis Tsakalides, "On the impact of tensor completion in the classification of undersampled hyperspectral imagery," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1975–1979.

[18] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al., "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[20] François Chollet et al., "Keras," `https://keras.io`, 2015.