NovaSearch at ImageCLEFmed 2016 Subfigure Classification Task

David Semedo and João Magalhães

NOVA LINCS, Department of Computer Science
Faculty of Science and Technology
Universidade NOVA de Lisboa, Portugal
`df.semedo@campus.fct.unl.pt`, `jmag@fct.unl.pt`

Abstract. This paper describes the NovaSearch team participation in the ImageCLEF 2016 Medical Task in the subfigure classification subtask.
Deep learning techniques have proved to be very effective in automatic representation learning and classification tasks with general data. More specifically, convolutional neural networks (CNNs) have surpassed human-level performance in the ImageNET classification task, making them a promising model for the task of medical modality classification. We assess how each model behave when dealing with medical images, by developing three different models, with different depths and components, and analyse the impact of these factors in the performance. One of the key ingredients for the effectiveness of CNNs (and deep learning in general) is the use of large amounts of data for training. This subtask scenario is completely different, due to the small size of the dataset, implying a significant risk of overfitting. We apply state-of-the-art techniques developed to reduce overfitting in these networks to our models and evaluate their effectiveness. Our best model achieves 65.31% accuracy on the test set using only the training data provided.

Keywords: Medical Modality Classification, Deep Learning, Convolutional Neural Networks

1 Introduction

This paper describes the NovaSearch team submissions, from the Faculty of Science and Technology of Universidade Nova de Lisboa, to the ImageCLEF 2016 [16] Medical task [2]. This task consists of five subtasks: compound figure detection, multi-label classification, figure separation, subfigure classification and caption prediction. We addressed the subfigure classification task, which aims to classify medical images within a given set of modalities.

We were interested in evaluating deep learning methods, namely convolutional neural networks (CNNs), in the specific scenario of this subtask. More concretely, we wanted to evaluate their effectiveness when dealing with medical images, which possess very distinct characteristics compared to general images, and for which they have proved lately to be very effective.

The remainder of this paper is organised as follows. In section 2 we describe our approach. More specifically, we present and discuss in detail the models and techniques used in our submitted runs. The configuration and results of each run are presented and discussed in section 3. Additionally, we compare our classifier results with the results achieved by the winning team. Finally, we draw conclusions and present future work perspectives on section 4.

2 Methodology

Regarding medical diagnostic and research, effective medical image retrieval systems (MIRS) can be a valuable tool for aiding clinicians. Furthermore, images in biomedical literature usually are compound figures with multiple panels each, with an image from a given modality (e.g. in figure 1), promoting human readability/interpretation by grouping correlated images, but making automatic retrieval more difficult.

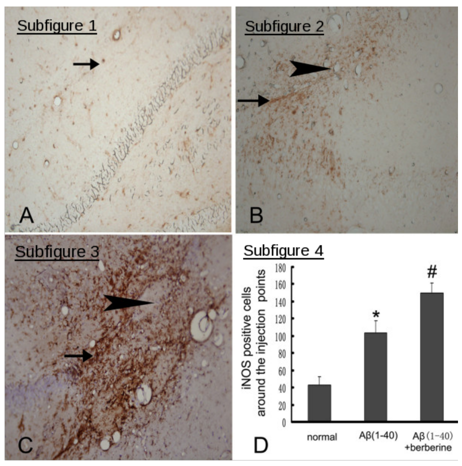

Fig. 1. Example of compound image with 4 subfigures, from a biomedical article. Adapted from [17].

ImageCLEFmed promotes research in this direction, by proposing a set of tasks, which result from splitting the problem of building a MIRS. Concretely, an effective MIRS must be capable of identifying and separate compound figures from biomedical articles and classify each subfigure with a given modality.

Knowing the modality of a medical image has been shown to be important to improve the performance of MIRS [8]. The subfigure task aims at classifying each subfigure, from a collection of figures from compound images found in biomedical articles from PubMed Central[1], into 30 modalities structured hierarchically (the hierarchy is defined in [7]).

We tackled the subfigure task with a deep neural network classifiers. Deep neural networks have recently achieved very good results in representation learning and classification of images [9, 11, 14, 5, 4]. Regarding image classification, Convolutional Neural Networks (CNNs), which are a type of neural network suitable for images capable of learning automatically high-level representations by definition, have proved to be very effective in this task. Since this task involves classifying images, CNNs were a natural choice.

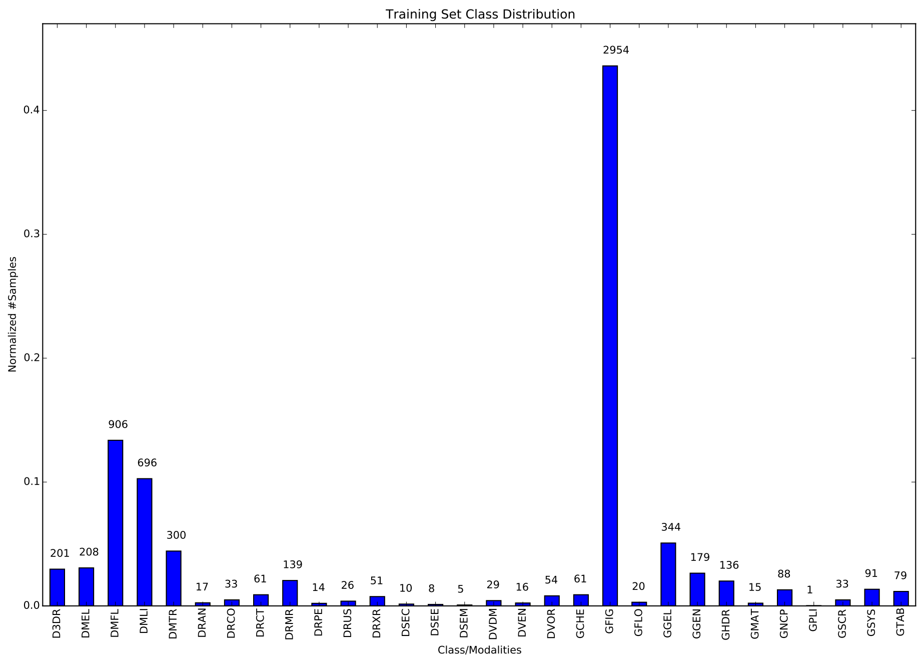

Fig. 2. Training set class distribution. The absolute number of examples from each modality is represented at the top of each bar.

2.1 Dataset

The provided training dataset consists of a collection of 6776 labelled medical images. Figure 2 depicts the class distribution. It is clear from the figure that

[1] PubMed Central (http://www.ncbi.nlm.nih.gov/pmc/) is a subset of PubMed, one of the major research databases containing scholarly articles that have been published within the biomedical and life sciences journal literature.

the dataset is highly unbalanced, specially towards the CFIG (Statistical figures, graphs, charts) class which represents $\approx 42\%$ of the examples.

Deep learning methods have achieved great results using very large datasets like the ImageNet challenge dataset [10] which contains roughly 1.2 million images from 1000 classes with approximately 1000 images per class. However, in the present task, the size of the dataset imposes a big challenge due to its size, specially for deep learning methods, which tend to overfit with small datasets. The fact that the dataset is highly unbalanced, making this a very challenging task.

We intend to evaluate how CNNs behave in such a scenario, which is clearly different from the scenarios in which they excelled, and experiment state-of-the-art techniques used to improve their performance.

2.2 Convolutional Neural Networks

Convolutional neural networks are a type of neural network which have very interesting characteristics. These networks are able to automatically learn high-level and hierarchical representations from data which eliminates the necessity of selecting a good set of low-level and high-level features to describe each image by hand.

By definition, CNNs make some assumptions (which are correct in general) regarding the stationarity of statistics and locality of pixel dependencies, allowing for a reduction in the number of connections and consequently the number of parameters to learn [1, 9]. Through depth and breadth one can control their capacity of identifying high-level data representations and relationships between the input and the output. From this reduction in the number of parameters and with current GPUs processing power, the task of training deep convolutional networks is feasible. The motivation for building and training deep CNNs is based on the fact that as the number of layers (depth) of the network increases, so the capacity of detecting more high-level details does, in principle.

2.3 Dealing with Unbalanced and Small Datasets

Despite the fact that this task involves classifying medical images, which have a set of exclusive characteristics of their own, the challenges of working with unbalanced and small datasets is a general problem in machine learning. In this section we describe a set of techniques that we applied to our models in order to take into account the training dataset characteristics.

Large networks tend to overfit. Traditional techniques to solve the overfitting problem consist of stopping the training procedure as soon as the validation error starts increasing (early-stopping) or in using regularisation techniques like adding an extra term in the function to be minimised or limiting the complexity of the model [1].

Recently, a technique named *Dropout* was proposed in [12] which has shown to be very effective in reducing overfitting in large networks, leading to performance improvements. Dropout can be interpreted as a stochastic regularization technique which essentially consists in randomly dropping neurons and their respective connections during training. The idea is to prevent neurons from co-adapting too much to data, making overfitting less likely.

Another technique we used to address overfitting and the class imbalance problem was data augmentation. More concretely, we performed real time data augmentation, in the sense that new images are generated from sample images at each training batch construction. The following operations are applied randomly to sampled images: horizontal/vertical shifting and horizontal/vertical flipping.

For classes which have few examples, the network may not be able to learn discriminative properties and may fail to generalise. Furthermore, in section 2.1 we pointed out that some classes (e.g. CFIG) dominate the dataset, which means that the majority of the weight updates during the training phase will be based in examples of these classes. It is therefore important to avoid focusing learning in some classes or we take the risk that the network will classify classes with few examples into a dominating class. We attempt to address this problem by modifying the loss function and making it a weighted loss function. We assign weights to each class such that it is worse to misclassify an image from a class with few examples.

Let P_w be the ideal number of examples of each class c_i assuming that the dataset is perfectly balanced. Then, P_w is obtained as follows:

$$P_w = \frac{N}{NC} \tag{1}$$

where N is the dataset size and NC is the total number of classes. The weight w_i for a class c_i is computed using the following expression:

$$w_i = \frac{P_w}{|S_i|} \tag{2}$$

where $|S_i|$ is the cardinality of the set of samples of class c_i from the training dataset. After computing the weights, each w_i is normalised such that $w_i \in [0, 1]$. By computing the weights with the expression above, we somehow simulate training with a perfectly balanced dataset. This is based on the fact that the values of the gradient to be back-propagated will be amplified for classes with few examples and reduced for the remaining ones.

In our first experiments these technique did not yield any performance improvement. In fact it deteriorated the performance of our models. After some experiments we concluded that the problem was that the network was severely misclassifying images from dominating classes due to the fact the respective weights are very small. To address this issue we introduced a lower bound on the weights values.

2.4 CNN Developed Models

Our submitted runs are based on essentially three different CNN models which we describe in the following sections and that will denoted from now on by VGG_1-CNN, VGG_2-CNN and $PReLU$-CNN.

For all the network models, the input consists of an 224x224 matrix.

The last layer of the networks has the $softmax$ activation function with dimension 30 (number of modalities of the medical classification subtask). The $softmax$ function enforces the constraint that outputs must lie between 0 and 1, and the sum of all output values is equal to 1, allowing the outputs of the network to be interpreted as posterior probabilities for categorical target variables.

For the special case of medical images, and considering the ImageCLEFmed hierarchy, images may share characteristics which in principle can help the classifier discriminate between modalities (e.g. certainly all Radiology images share some characteristics). However, taking this into account would require a different and possibly more complex approach. By using $softmax$ we do not model the fact that the modalities are structured hierarchically, and the final model is not an hierarchical classifier, but an unstructured one. This relaxation has proved to be effective while not increasing the model conceptual complexity [9, 11, 14].

VGG-like models Both VGG_1-CNN and VGG_2-CNN models are inspired in the VGG model proposed in [11] which achieved top results in the ImageNET Large Scale Visual Recognition Challenge 2014 (ILSVRC 2014).

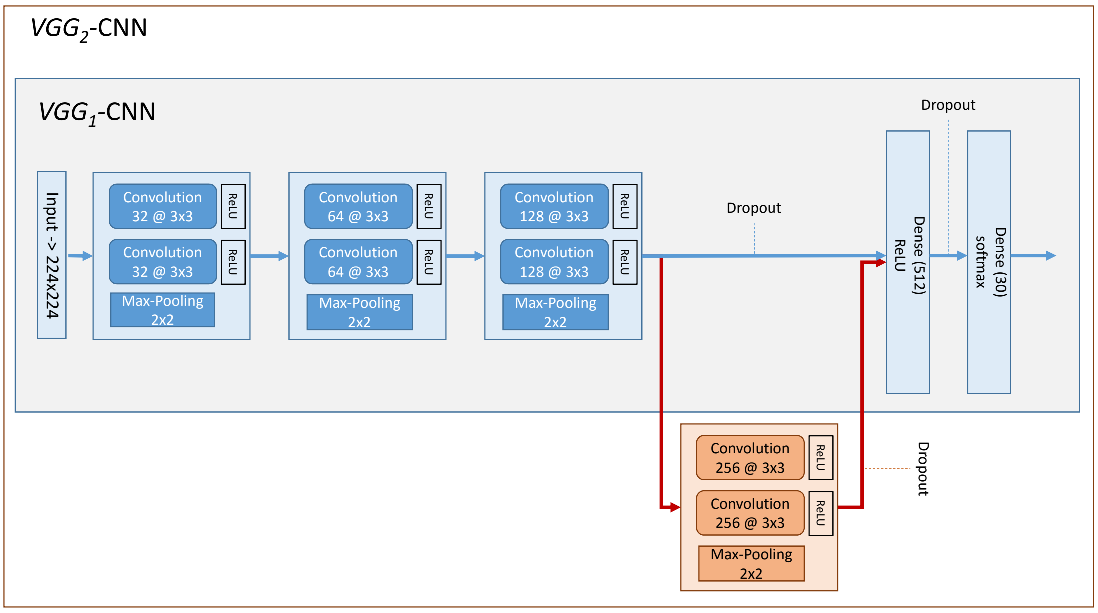

Fig. 3. VGG_1-CNN and VGG_2-CNN models architecture. The shaded area represents the architecture of model VGG_1-CNN and the outer box represents the architecture of model VGG_2-CNN.

This model consists of a deep network with several identically parametrised convolutional layers using small receptive fields (3x3). By using small receptive fields a deeper network can be achieved while keeping the number of parameters equivalent to more shallow networks. Max-pooling is performed after some convolutional layers. For all hidden layers the activation function used is the Rectified Linear Unit [9] (ReLU) which has shown to yield faster training since it does not saturate as the *tanh* and *sigmoid* activation functions.

We performed some modifications mainly due to computational constraints but also due to overfitting. The original best performing models have 16 and 19 weight layers respectively, and roughly 140 million parameters each. This induces large training times and requires a large dataset such that the network can effectively learn all the parameters and generalize. The medical subfigures dataset is very small compared to the ILSVRC dataset and overfitting becomes a serious issue. Therefore, we reduced the number of convolutional layers and removed one of the fully connected layers. Additionally, we reduced the number of channels in each convolutional layer, in a way that the new values are still proportional to the original model. All the remaining components and characteristics of the architecture like the max-pooling (and its shape), dropout and activation functions are preserved.

Figure 3 presents both model architectures. The difference between both models is in the depth (VGG_2-CNN has greater depth). Just like in the original VGG model, dropout is used between the last convolutional layer and the first fully connected layer, and between this and the *softmax* layer.

PReLU with Batch Normalization Model

The ReLU activation function has a reduced likelihood of suffering from gradient vanishing. This is due to the fact that derivatives through the rectifier remain large whenever the unit is active, unlike other activation functions like *sigmoid* and *tanh*. However, when the value of the argument of the ReLU function is not positive, the gradient will be 0 and it will not be able to learn.

Recently, the *Parametric ReLU* (PReLU) was proposed in [5] which consists in changing the slope of the ReLU function for inputs in $\mathbb{R}_{\leq 0}$, by multiplying it by a coefficient a_i. The coefficient a_i is treated as a learnable parameter. The final expression is:

$$f(x_i) = max(0, x_i) + a_i min(0, x_i) \tag{3}$$

With these expression, when the inputs of the function are in $\mathbb{R}_{\leq 0}$, the gradient will not be 0.

In deep neural networks, despite the fact that in pre-processing steps the network inputs are normalised (shifted to zero-mean and unit variance), the distribution of each layer's inputs changes during training in function of the parameters of the previous layers. This problem is referred as *internal covariate shift* [6]. Consequently, one needs to use lower learning rates and carefully initialise the parameters, slowing down training.

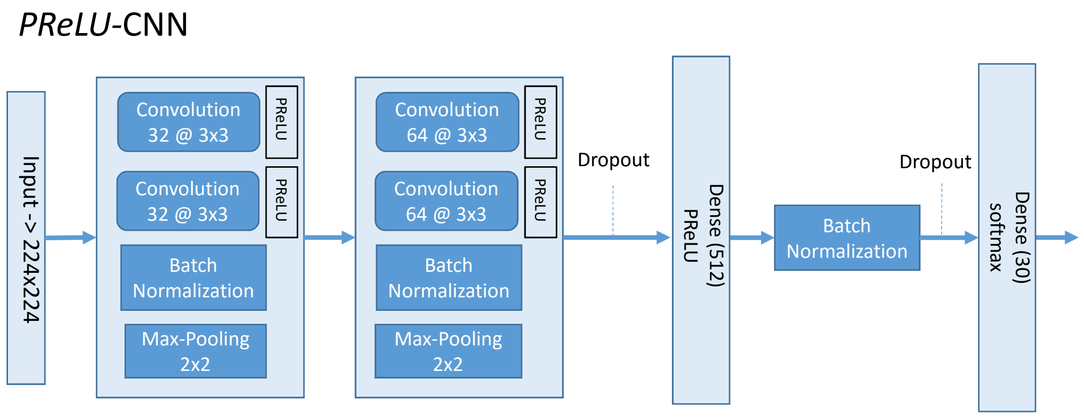

Fig. 4. *PReLU*-CNN model architecture.

To address these problems in [6] the Batch Normalisation technique is proposed. It essentially consists of performing mini-batch normalisation at intermediate nodes of the network.

We developed a CNN model (*PReLU*-CNN) which uses both PReLU's and batch normalisation techniques. The model can be seen in figure 4. Both these techniques increase the computational requirements during training, therefore, we had to reduce the network depth to be able to train the network in feasible time for the submission.

Networks Hyperparameters
The three proposed models are trained using stochastic gradient descent with Nesterov momentum [13] and learning rate decay is applied. The training hyperparameters values are the same and can be seen in Table 1. These values were defined empirically.

Table 1. Hyperparameter values shared among all models.

Hyperparameter	**Value**
Batch size	20
Learning Rate	0.005
Momentum	0.9
Learning Rate Decay	1×10^{-6}

Our model implementation allows enabling or disabling dropout. In section 3 in which we describe our submitted runs, we point out which ones use it. The probability of dropping units between the last convolutional layer and the first fully connected layer is 0.25, between the first fully connected layer and the *softmax* layer is 0.5. Xavier initialisation algorithm [3] was used for initialising all the weights of our models.

2.5 Data Pre-processing

As stated in the previous section, despite the fact that our models architecture is based on the VGG model which takes as input an RGB image (the image matrix has one dimension for each color channel) we only used one channel, to reduce the training time. All images are resized to 224x224. Additionally, the elements of each image matrix were scaled to the interval $[0, 1]$.

We split training data into 70/30 training and validation splits, respectively.

3 Submitted Runs and Results

All our models were trained using exclusively the training dataset provided. The dataset has the same sample images as the set defined by the merge of the training and test sets from the 2015 edition of the subfigure task. Therefore, for fixing parameters we used the 2015 training/test split since it will have a similar distribution.

Training CNNs is a very computationally demanding task. Recently several efficient libraries that take advantage of GPU computing capabilities to speedup computations, have been developed. We implemented our models using Keras[2] along with Theano [15] (Python) libraries. Keras is a neural networks python library that features a rich set of components for developing neural networks models and Theano is an efficient numerical computation library with GPU support.

We submitted four runs for the subfigure classification task. Table 2 depicts each run configuration and the epoch in which the lowest validation error was obtained.

Table 2. Configurations of all the submitted runs regarding the model used, the use of dropout, the number of epochs and the training iteration with lowest validation error (#Best Epoch.).

Run Name (Number)	**Model**	**Dropout**	**#Epochs**	**#Best Epoch.**
SC_NOVASearch_cnn_8_vgglike (1)	VGG_1-CNN	True	500	314
SC_NOVASearch_cnn_10_vgglike (2)	VGG_2-CNN	False		438
SC_NOVASearch_cnn_10_dropout_vgglike (3)	VGG_2-CNN	True		470
SC_NOVASearch_cnn_prelu (4)	$PReLU$-CNN	True		482

For all runs we fixed the number of epochs to 500. Hyperparameters are also shared among all runs (see section 2.4) apart from dropout on run #2 which

[2] Keras python library: https://github.com/fchollet/keras

is not enabled. We verified that except for run #1 which achieves the lowest validation error at roughly half of the total number of epochs, in remaining runs it is achieved in late epochs.

The results (accuracy on validation and test datasets) for each run are presented in table 3. Our models are not in par with the best results achieved for the task in the current edition.

A first observation is that our models performance both on validation and test sets are very similar, which indicates that training splits were suited, i.e., have an identical distribution as the test set. The deeper model with dropout (run #3) achieved the best result (65.31% accuracy) taking ≈ 6 hours to train. The exact same model with the same configuration but without dropout (run #2) took almost 6 hours to train and achieved approximately less 3% accuracy. Therefore, it is clear that dropout does contribute to performance improvements.

Table 3. Results achieved with each of the submitted runs. Accuracy on validation (Val. Acc.) and test (Test. Acc.) data is shown. For comparison, we also present the global best visual and mixed submissions results achieved in the subtask.

Run Name (Number)	**Val. Acc. (%)**	**Test. Acc. (%)**
SC_NOVASearch_cnn_8_vgglike (1)	70.5	65.17
SC_NOVASearch_cnn_10_vgglike (2)	68.4	63.29
SC_NOVASearch_cnn_10_dropout_vgglike (3)	70.7	**65.31**
SC_NOVASearch_cnn_prelu (4)	65.7	63.80
Best Visual Run	-	85.38
Best Mixed Run	-	88.43

From the results table, we can also see that the run #1 got almost as good results as run #3 despite the fact that its model is not as deep, taking ≈ 4 hours to train. This is an indication that we are probably observing overfitting on both models.

To assess this issue we focused on the best performing run model (#3). We plotted in the y-axis the error obtained in the training and validation splits across the 500 epochs (x-axis) for the model used in our best run (#3). The plot is shown in figure 5. The error curves are very noisy but it is visible that the model is able to gradually converge towards smaller errors until it starts overfitting. We believe that using a low learning rate (0.005) was crucial to achieve this convergence.

Although this model uses both dropout and data augmentation techniques, which help regularising the model, it is clear that overfitting behaviour can be verified from epoch 100. The training dataset class skewness and size contribute to this behaviour. Therefore, a different approach than the one we used for this subtask is needed to deal with overfitting and help the model generalise. The remaining models also suffer from overfitting.

Regarding run #4 which uses the *PReLU*-CNN model, it achieved the third best result on the test set, with an accuracy of 63.8%, outperforming model

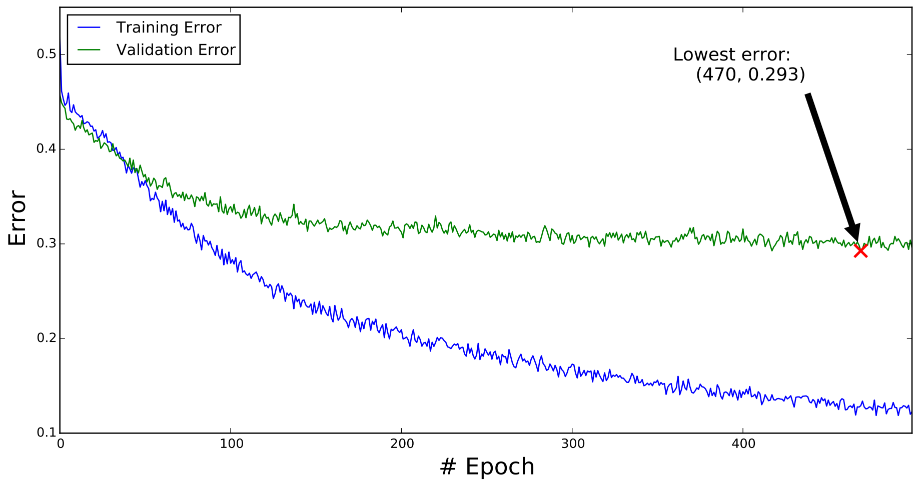

Fig. 5. Training and validation error of the model used in run *SC_NOVASearch_cnn_10_dropout_vgglike* during training.

VGG_1-CNN, despite being a less deeper model and taking significantly more time to train ($\approx$ 21 hours). This is very likely a consequence of using both PReLUs and batch normalisation techniques, although in order to confirm this fact, additional experiments using a network with the same depth as the ones from models VGG_1-CNN and VGG_2-CNN are required such that models are comparable.

4 Conclusions and Future Work

In this paper we developed three deep learning models for tackling the subfigure medical modality classification subtask. We performed a set of experiments using CNNs, which excel with general images in scenarios where large amounts of data are available but not with small and highly unbalanced datasets. Our objective was to assess their effectiveness in classifying medical images with respect to this task challenging scenario. A set of state-of-the-art techniques for dealing with overfitting in CNNs were used in the models developed. Our best model achieved an accuracy of 65.30% using only training data provided, which we believe its a good result given the challenge, making CNNs a promising tool for medical image modality classification.

We observed severe overfitting in our models despite our efforts in reducing it, therefore, a different approach is needed. Namely, one technique would be to modify the batch construction and ensure that batches are constructed by sampling from the dataset with all the 30 classes being uniformly distributed. The key idea is that the probability of each batch having a sample from a given class is the same for all classes, possibly reducing overfitting on the most dominant classes.

A better hyperparameters tuning protocol must be used so that we understand the true influence of each hyperparameter in the results achieved. Additionally, we may be losing important properties of images by not using all the three color channels, therefore our experiments should be repeated with the input of all the models being an RGB image.

Acknowledgements This research was partially supported by the project NOVA LINCS Ref. UID/CEC/04516/2013. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

References

1. Ian Goodfellow Yoshua Bengio and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.
2. Alba García Seco de Herrera, Roger Schaer, Stefano Bromuri, and Henning Müller. Overview of the ImageCLEF 2016 medical task. In *Working Notes of CLEF 2016 (Cross Language Evaluation Forum)*, September 2016.
3. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.
4. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *arXiv preprint arXiv:1506.01497*, 2015.
5. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1026–1034, 2015.
6. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 448–456, 2015.
7. Jayashree Kalpathy-Cramer, Alba García Seco de Herrera, Dina Demner-Fushman, Sameer Antani, Steven Bedrick, and Henning Müller. Evaluating performance of biomedical image retrieval systems – an overview of the medical image retrieval task at ImageCLEF 2004–2014. *Computerized Medical Imaging and Graphics*, 2014.
8. Jayashree Kalpathy-Cramer and William R. Hersh. Automatic image modality based classification and annotation to improve medical image retrieval. In *MEDINFO 2007 - Proceedings of the 12th World Congress on Health (Medical) Informatics - Building Sustainable Health Systems, 20-24 August, 2007, Brisbane, Australia*, pages 1334–1338, 2007.
9. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems, NIPS 2012*, pages 1097–1105, 2012.

10. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
11. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, abs/1409.1556, 2015.
12. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.
13. Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 1139–1147, 2013.
14. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.
15. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.
16. Mauricio Villegas, Henning Müller, Alba García Seco de Herrera, Roger Schaer, Stefano Bromuri, Andrew Gilbert, Luca Piras, Josiah Wang, Fei Yan, Arnau Ramisa, Emmanuel Dellandrea, Robert Gaizauskas, Krystian Mikolajczyk Joan Puigcerver, Alejandro H. Toselli, Joan-Andreu Sánchez, and Enrique Vidal. General Overview of ImageCLEF at the CLEF 2016 Labs. Lecture Notes in Computer Science. Springer International Publishing, 2016.
17. Feiqi Zhu and Caiyun Qian. Berberine chloride can ameliorate the spatial memory impairment and increase the expression of interleukin-1beta and inducible nitric oxide synthase in the rat model of alzheimer's disease. *BMC neuroscience*, 7(1):78, 2006.