Towards Representation Learning for Biomedical Concept Detection in Medical Images: UA.PT Bioinformatics in ImageCLEF 2017

Eduardo Pinho, João Figueira Silva, Jorge Miguel Silva, and Carlos Costa

DETI - Institute of Electronics and Informatics Engineering of Aveiro
University of Aveiro, Portugal
`{eduardopinho, joaofsilva, jorge.miguel.ferreira.silva, carlos.costa}@ua.pt`

Abstract. Representation learning is a field that has rapidly evolved during the last decade, with much of this progress being driven by the latest breakthroughs in deep learning. Digital medical imaging is a particularly interesting application since representation learning may enable better medical decision support systems. ImageCLEFcaption focuses on automatic information extraction from biomedical images. This paper describes two representation learning approaches for the concept detection sub-task. The first approach consists of k-means clustering to create bags of words from SIFT descriptors. The second approach is based on a custom deep denoising convolutional autoencoder. A set of perceptron classifiers were trained and evaluated for each representation type. Test results showed a mean F1 score of 0.0488 and 0.0414 for the best run using bags of words and the autoencoder, respectively.

Keywords: ImageCLEF · Representation Learning · Deep Learning · Multimedia Retrieval

1 Introduction

Representation learning has been a rapidly evolving field during the last decade [2]. The discovery of more powerful representation learning techniques opens up tremendous prospect for semi-supervised and unsupervised decision systems, and further unlocks the potential of content-based image retrieval (CBIR). A significant part of this progress comes as a consequence of the latest breakthroughs in deep learning. This extensive use of deep learning is no exception in health informatics, including medical imaging, where a vast range of use-cases have been tackled, and multiple solutions have relied on deep learning for such purposes [15]. Due to the inherent nature of medical imaging datasets, which are scarce and both frequently class-imbalanced and non-annotated, the rapid developments in deep learning and representation learning pose particular interest for the medical field, since such developments may enable better concept representation of digital medical imaging.

Representation learning, sometimes called *feature learning*, is often defined as learning a function that transforms the available data samples into a representation that makes other machine learning tasks easier to approach. *Feature extraction* is the related concept of obtaining these representations. This can be achieved using a wide range of approaches, such as k-means clustering, sparse coding [12] and Restricted Boltzmann Machines (RBMs) [8]. More recently, with the shift of interest leading to a focus on deep learning techniques, approaches based on autoencoders [16] have also been developed.

The long-running ImageCLEF initiative has introduced the **caption** challenge for 2017 [10], aiming for the automatic information extraction from biomedical images. This challenge is divided into two sub-tasks: *concept detection* and *caption prediction*. The *concept detection* sub-task [5] is the first part of the caption prediction task, in which the goal is to automatically recognize certain concepts from the UMLS vocabulary [3] in biomedical images. The obtained list of concepts is then used in the *caption prediction* sub-task, where a small human-readable description of the image must be generated.

For this challenge, we have hypothesized that a sufficiently powerful representation of images would enable a medical imaging archive to automatically detect biomedical concepts with some level of certainty and efficiency, thus improving the system's information retrieval capabilities over non-annotated data.

This paper presents our solution proposal for the *concept detection* sub-task, and describes our methods of image feature extraction for the purpose of biomedical concept detection, followed by their evaluation under the ImageCLEF 2017 challenge.

2 Methods

This task was accompanied with three data sets containing various images from biomedical journals: the *training set* (164614 images), the *validation set* (10000 images) and the *testing set* (10000 images). Only the first two included the list of concepts applicable to each image, whereas the testing set's annotations were hidden from the participants.

In order to evaluate the imposed hypothesis of a middle-level representation for medical images, we have addressed the concept detection task in two phases. First, two feature extraction methods for image representation were chosen and built:

- In Section 2.1, as a classical approach, bags of visual words were used as image descriptors, obtained from the clustering of visual keypoints;
- In Section 2.2, a deep convolutional sparse autoencoder was trained and features were extracted from its bottleneck vector.

Secondly, the two representations were validated by training fast classifiers over the new representations, in Section 2.3.

2.1 Bags of Visual Words

Without resizing or preprocessing the images, Scale Invariant Feature Transform (SIFT) keypoint descriptors [13] were extracted from all three datasets. An OpenCV [4] implementation was used for SIFT keypoint extraction and descriptor computation. Each image could yield a variable number of descriptors of size 128. In cases where the SIFT algorithm did not retrieve any keypoints, the algorithm's parameters were adjusted to loosen edge detection criteria.

From the training set, 500 files were randomly chosen and their respective keypoints collected to serve as template keypoints. A visual vocabulary (codebook) of size $k = 1000$ was then obtained by performing k-means clustering on all template keypoints and retrieving the centroids of each cluster, yielding an ordered list of 1000 vectors of fixed size $\mathcal{V} = \{V_i\}$.

Once a visual vocabulary was available, we constructed an image's bag of visual words (BoW) by determining the closest visual vocabulary point and incrementing the corresponding position in the BoW for each image keypoint descriptor. In other words, for an image's BoW $B = \{o_i\}$, for each image keypoint descriptor d_j, o_i is incremented when the smallest Euclidean distance from d_j to all other visual vocabulary points in V is the distance to V_i. Finally, each BoW was normalized so that the sum of all elements in a BoW equals 1. We can picture the bag of visual words as a histogram of visual descriptor occurrences, which can be used for representing visual content in CBIR.

2.2 Sparse Autoencoder

A deep convolutional neural network was designed for the unsupervised extraction of visual features from biomedical images (Figure 1). It is a quasi-symmetrical autoencoder with a series of encoding and decoding blocks (respectively named henceforth ci and di for $i \in \{1, 2, 3\}$) with shared weights. A sparse latent code representation of dimensionality 10000 (ten thousand) lies in the middle. As a denoising autoencoder, its goal is to learn the pair of functions (E, D) so that $x' = D(E(\tilde{x}))$ is closest to the original input x, where $\tilde{x}$ is a slightly corrupted version of x. The aim of making E a function of $\tilde{x}$ is to force the process to be more stable and robust, thus leading to representations of higher quality [16].

2.2.1 Encoder / Decoder Specification

Each encoder block ci is composed of two sequences of 2D convolution components, where each convolution is followed by batch normalization [9] and Rectified Linear Unit (ReLU) activations.

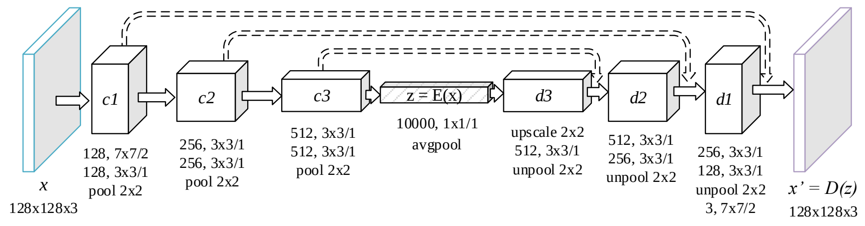

Fig. 1: Schematic representation of the autoencoder. The $c\#$ and $d\#$ blocks contain convolutional and pooling/unpooling layers in the indicated order. The dashed arrows represent the transfer of pooling indices to guide the unpooling layers.

Then, a 2D max-pooling layer with a 2x2 kernel is added. The first convolutional layer of $c1$ relies on a kernel (filter) of size 7x7, whereas the remaining convolutions have a kernel of size 3x3. The exact numbers of kernels in each convolutional layer are shown in Figure 1, starting with 64 filters and duplicating upon each new encoder block. The layers are also described with greater detail in Table 1. Instead of max-pooling in the third block, a 1x1 kernel convolution is performed, followed by global average pooling and a ReLU activation, yielding the code tensor z. The 1x1 convolution followed by global average pooling behaves similarly to a fully connected network, with the advantage of making the network invariant to input dimensions.

Tbl. 1: A tabular representation of the encoder layers' specifications. The Details column may include the normalization and activation layers that follow a layer (where BN stands for batch normalization and $\mathrm{ReLU}(x) = \max(0, x)$).

Layer	Kernels	Size/Stride	Details
conv1	128	7x7 /2	BN + ReLU
conv2	128	3x3 /1	BN + ReLU
pool2	N/A	2x2 /2	
conv3	256	3x3 /1	BN + ReLU
conv4	256	3x3 /1	BN + ReLU
pool4	N/A	2x2 /2	
conv5	512	3x3 /1	BN + ReLU
conv6	512	3x3 /1	BN + ReLU
pool6	N/A	2x2 /2	
conv7	10000	1x1 /1	BN, linear activation
avgpool	N/A	8x8	ReLU, $h = E(\tilde{x})$

The decoding blocks replicate the encoding process in inverse order (Table 2). It starts with an upsampling of the code to the same three-dimensional feature shape as the encoder's final convolutional layer. Convolutions in these blocks are transposed (also called *fractionally-strided convolution* in literature, and *deconvolution* in a few other papers). Furthermore, the unpooling layer is a *guided* unpooling operation, in which the exact position of the encoder's max-pool activation is replicated in the same position in the corresponding decoding phase. This is achieved by passing the switch indices from the encoder's max-pooling layers, as in [14].

Tbl. 2: A tabular representation of the decoder layers' specifications. The Details column may include the normalization and activation layers that follow a layer, as well as information about weight sharing.

Layer	Kernels	Size/Stride	Weight share	Details
upsample	N/A	8x8		
dconv7	512	7x7 /2		BN + ReLU
unpool6	N/A	2x2 /2		Guided by "pool6"
dconv6	512	3x3 /1	conv6	BN + ReLU
dconv5	256	3x3 /1	conv5	BN + ReLU
unpool4	N/A	2x2 /2		Guided by "pool4"
dconv4	256	3x3 /1	conv4	BN + ReLU
dconv3	128	3x3 /1	conv3	BN + ReLU
unpool2	N/A	2x2 /2		Guided by "pool2"
dconv2	127	3x3 /1	conv2	BN + ReLU
dconv1	3	7x7 /2	conv1	Linear activation, $x' = D(h)$

The autoencoder was designed with this symmetry to enable the encoder and decoder pair to share the weights of the kernels. These weights were initialized as in [7], and were shared in a way that matrix W_i is used by the ith convolution layer in the encoder (counting from the left) and by the same transpose convolution number when counted from the right. The only exception lies in the convolutional layer pair closest to the latent code, which do not share any parameters. The layers' bias parameter and batch normalization coefficients are also not shared among the two parts of the network.

2.2.2 Preprocessing and Augmentation

Training samples were obtained through the following process: images were resized so that its shorter edge size was 160 pixels. Afterwards, the sample was augmented using random square 128 pixel-wide random crops (of 9 possible kinds of crops: 4 corners, 4 edges and center). Validation and test images were simply resized to fit the 128x128 dimensions. For all cases, images' pixel RGB values were normalized with the formula $n(v) = v/128.0 - 1$, thus sitting in the range [-1, 1]. In the training phase, a Gaussian noise of standard deviation 0.05 was applied over the input, yielding the noisy sample $\tilde{x}$.

2.2.3 Network Training Details The network was trained through stochastic gradient descent, by minimizing the mean squared error between the input x and the output x', using the Adam optimizer [11]. A sparse representation was achieved with two mechanisms: first, since the final encoding activation is ReLU, negative outputs from the previous layer are zeroed. Second, an absolute value penalization was applied to z, thus adding the extra minimization goal of keeping the code sum small. The final decoder loss function was therefore:

$$\mathcal{L}(E, D) = \frac{1}{2r} \sum_{i=0}^{\mathrm{r}} (x_i - x'_i)^2 + \Omega(z)$$

where

$$\Omega(z) = s \times \max\left(0, \sum_{z_i}^{z} z_i - t\right)$$

is the sparsity penalty function, r $= 128 \times 128$ is the number of pixels in the input images, and x represents the original input without synthesized noise. t and s are, respectively, the penalization threshold and the sparsity coefficient hyperparameters, which we left defined as $t = 1$ and $s = 0.0001$. At the final training iteration, 73% of extracted features from the training set were zeros on average. Sparser representations are possible by adjusting these hyperparameters.

The model was trained over 20600 steps, which is approximately 8 epochs, with a mini-batch size of 64. The autoencoder's loss evolved according to Figure 2. The base learning rate was 0.005, but the first 50 steps used a *"warm-up"* learning rate of 0.001 to prevent the initial loss values from producing extreme activations, which could make the network harder to converge (a similar procedure was done in [6]). The learning rate was multiplied by 0.2 every 5140 steps ($\pm$ two epochs), to facilitate convergence.

TensorFlow [1] with GPU support was used to train the neural network and retrieve the latent codes of each image in the three datasets. A custom version between 1.1.0 and 1.2.0 was used (specifically, the commit hash `163b1c078d` in the official GitHub repository[1]) in order to have early access to the gradient implementation of the max-pooling layer with arg-max propagation. TensorBoard [1] was used during the development for monitoring and visualization. Training took approximately 22 hours to complete on one of the GPUs of an NVIDIA Tesla K80 graphics card in an Ubuntu server machine.

[1] https://github.com/tensorflow/tensorflow

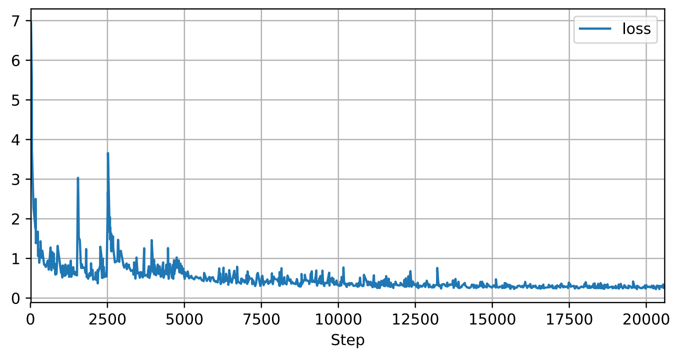

Fig. 2: Autoencoder loss evolution over the steps of the training process.

2.3 Concept Classification Perceptron

For each representation learned, a simple classification model was applied to the resulting features. Aiming for low complexity and classification speed, the three sets of classifiers were perceptrons trained using stochastic gradient descent over the training set.

In the first run submitted (*#1*), the learning rate was configured to slowly decay during the training process ($\mu(t) = \frac{1}{\alpha t+1}$) where an L_2 regularization term with coefficient $\alpha = 0.005$ was included. The last two runs submitted (*#2* and *#3*) had slight changes in hyperparameters, which appeared to yield better outcomes when considering a small sample of biomedical concepts. A constant learning rate of 1.0 was used instead, and classes were balanced so that the weight of each class was the inverse of its frequency in the set, thus aiming to penalize the presence of false negatives through higher losses. While run *#2* used the SIFT bags of words, run *#3* used the latent features from the autoencoder.

Due to computational time constraints, the number of gradient descent epochs was relatively limited: 50 iterations for the SIFT bags of words and 20 for the autoencoder features. Biomedical concepts' identifiers were sorted by their total number of occurrences in the training set. With this list, the 1000 most frequent concepts in the training set were retrieved and a perceptron model was trained for each. With more time and computational resources, this approach can be linearly scaled to cover all labelled concepts.

Once trained, the models' performance was evaluated with the validation set, in terms of precision, recall, and mean F1 score. The same model was then used to predict the concept list of each image in the testing set. Other than this final prediction phase, neither the feature extractors nor the trained classifiers have ever been fed with samples from the testing set. Moreover, no other sources of data were used in the full process.

3 Results

Table 3 shows metrics obtained on the validation and test sets. The validation F1 metric, which was obtained from evaluating the model against the validation set, only assumes the existence of the 1000 most frequent concepts in the training set. Nonetheless, these metrics were deemed acceptable for a quantitative comparison among local runs, and have indeed defined the same ranking order as the final test metrics'.

Tbl. 3: Results obtained from the three submissions to the ImageCLEF 2017 concept detection task.

#	ImageCLEF Run File	Kind	Val. Recall	Val. F1	Test F1
1	DET_0503192045.txt	SIFT-BoW	0.166286	0.022324	0.0488
2	DET_0504234124-0.txt	SIFT-BoW	0.630096	0.020586	0.0463
3	DET_0505041340-0.txt	AE	0.553397	0.013764	0.0414

Unlike our previous expectations, the hyperparameter changes applied in runs *#2* and *#3* appeared to slightly cripple the model's performance, regardless of the extracted features. Applied modifications may also benefit from additional classifier training steps for a better convergence.

The autoencoder designed for this challenge exhibited the worst performance among submitted runs, while demanding more computational resources for training and feature extraction. However, given its known and recently well studied milestones in image analysis, this does not invalidate the use of deep convolutional neural networks in general. Rather, it suggests that some difficulty in model training for this domain was experienced, and proper concept detection learning would likely require further tweaking of the network's hyperparameters and more iterations.

Even after reducing the problem to a feature vector instead of the original input, the number of annotated concepts was too large (over 20 thousand) to train a good classifier for each concept. Higher numbers of annotated concepts imply higher computational costs, and available compute power was limited. In order to deal with this tradeoff, we decided to follow an approach of tackling the most frequent concepts, thus mitigating computational costs, but with the drawback of not providing any results for the remaining concepts.

4 Conclusion

This paper describes our proposal to solve the *concept detection* sub-task of the ImageCLEFcaption task, resting on the hypothesis that a sufficiently powerful representation of images can enable a medical imaging archive to automatically detect biomedical concepts with some level of certainty and efficiency. Results are presented for the three submitted runs, with the first two being based on a BoWs approach, whereas the third one is based on a deep convolutional sparse autoencoder.

First and foremost, it is important to mention that extreme variability was experienced in this challenge. Regarding obtained test results, a mean F1 score of 0.0488 and 0.0414 was obtained for the best run using BoW and for the autoencoder, respectively.

Attaining better feature representations is a major step that can have significant impact in the end results. Due to the major breakthroughs already originated by deep learning techniques, we believe that further research in representation learning should be carried out, as it will allow us to determine improved ways of training neural networks for this purpose, and evaluate a wider variety of feature learning solutions.

5 Acknowledgements

This work is financed by the ERDF - European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia. Eduardo Pinho is funded by the FCT under the grant PD/BD/105806/2014. João Figueira Silva is funded by the Research grant Project PTDC/EEI-ESS/68 15/2014 and Jorge Miguel Silva is funded by the Research grant Project CMUP-ERI/ICT/0028/2014-SCREEN-DR.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". In: (Mar. 2016). arXiv: 1603.04467. URL: http://arxiv.org/abs/1603.04467.

[2] Yoshua Bengio, Aaron Courville, and Pierre Vincent. "Representation learning: A review and new perspectives". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.8 (2013), pp. 1798–1828. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2013.50.

[3] Olivier Bodenreider. "The unified medical language system (UMLS): integrating biomedical terminology". In: *Nucleic acids research* 32.suppl 1 (2004), pp. D267–D270.

[4] Gary Bradski et al. "The OpenCV Library". In: *Doctor Dobbs Journal* 25.11 (2000), pp. 120–126.

[5] Carsten Eickhoff, Immanuel Schwall, Alba García Seco de Herrera, and Henning Müller. "Overview of ImageCLEFcaption 2017 - Image Caption Prediction and Concept Detection for Biomedical Images". In: *CLEF 2017 Labs Working Notes*. CEUR Workshop Proceedings. Dublin, Ireland: CEUR-WS.org <http://ceur-ws.org>, Sept. 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[7] 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.

[8] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets". In: *Neural computation* 18.7 (2006), pp. 1527–1554.

[9] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *arXiv preprint arXiv:1502.03167* (2015), pp. 1–11. ISSN: 0717-6163. DOI: 10.1007/s13398-014-0173-7.2. arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

[10] Bogdan Ionescu, Henning Müller, Mauricio Villegas, Helbert Arenas, Giulia Boato, Duc-Tien Dang-Nguyen, Yashin Dicente Cid, Carsten Eickhoff, Alba Garcia Seco de Herrera, Cathal Gurrin, Bayzidul Islam, Vassili Kovalev, Vitali Liauchuk, Josiane Mothe, Luca Piras, Michael Riegler, and Immanuel Schwall. "Overview of ImageCLEF 2017: Information extraction from images". In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction 8th International Conference of the CLEF Association, CLEF 2017*. Vol. 10456. Lecture Notes in Computer Science. Dublin, Ireland: Springer, Sept. 2017.

[11] Diederik P Kingma and Jimmy Lei Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations*. 2015. URL: https://arxiv.org/pdf/1412.6980.pdf.

[12] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. "Efficient sparse coding algorithms". In: *Advances in Neural Information Processing Systems* 19.2 (2006), pp. 801–808. ISSN: 10495258. DOI: 10.1.1.69.2112. arXiv: arXiv:1506.03733v1.

[13] David G Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110.

[14] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. “Learning deconvolution network for semantic segmentation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1520–1528.

[15] Daniele Ravi, Charence Wong, Fani Deligianni, Melissa Berthelot, Javier Andreu-Perez, Benny Lo, and Guang-Zhong Yang. “Deep Learning for Health Informatics”. In: *Biomedical and Health Informatics, IEEE Journal of* 21.1 (Jan. 2017), pp. 4–21. ISSN: 2168-2194. DOI: 10.1109/JBHI.2016.2636665. URL: http://ieeexplore.ieee.org/document/7801947/.

[16] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-antoine Manzagol. “Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion”. In: *Journal of Machine Learning Research* 11 (2010), pp. 3371–3408.