# Mimir: An Automatic Reporting and Reasoning System for Deep Learning based Analysis in the Medical Domain

Steven Alexander Hicks
Simula Research Laboratory, Norway
University of Oslo, Norway

Sigrun Eskeland
Department of Medical Research
Bærum Hospital
Vestre Viken Hospital Trust, Norway

Mathias Lux
Klagenfurt University, Austria

Thomas de Lange
Department of Transplantation
Oslo University Hospital, Norway
University of Oslo, Norway

Kristin Ranheim Randel
Cancer Registry of Norway

Mattis Jeppsson
ForzaSys AS, Norway

Konstantin Pogorelov
Simula Research Laboratory, Norway
University of Oslo, Norway

Pål Halvorsen
Simula Metropolitan Center for
Digital Engineering, Norway
University of Oslo, Norway

Michael Riegler
Simula Metropolitan Center for
Digital Engineering, Norway
University of Oslo, Norway

## ABSTRACT

Automatic detection of diseases is a growing field of interest, and machine learning in form of deep learning neural networks are frequently explored as a potential tool for the medical video analysis. To both improve the "black box"-understanding and assist in the administrative duties of writing an examination report, we release an automated multimedia reporting software dissecting the neural network to learn the intermediate analysis steps, i.e., we are adding a new level of understanding and explainability by looking into the deep learning algorithms decision processes. The presented open-source software can be used for easy retrieval and reuse of data for automatic report generation, comparisons, teaching and research. As an example, we use live colonoscopy as a use case which is the gold standard examination of the large bowel, commonly performed for clinical and screening purposes. The added information has potentially a large value, and reuse of the data for the automatic reporting may potentially save the doctors large amounts of time.

## CCS CONCEPTS

• **Computing methodologies** → *Machine learning*; *Video summarization*; • **Applied computing** → **Health informatics**;

## KEYWORDS

Deep learning, medical documentation, interpretable neural networks, automatic disease detection

Contact author's address: Michael Riegler, Simula Research Laboratory, Oslo, Norway, email: michael@simula.no .



*MMSys'18, June 12–15, 2018, Amsterdam, Netherlands*



ACM ISBN 978-1-4503-5192-8/18/06…$15.00
https://doi.org/10.1145/3204949.3208129

**ACM Reference format:**
Steven Alexander Hicks, Sigrun Eskeland, Mathias Lux, Thomas de Lange, Kristin Ranheim Randel, Mattis Jeppsson, Konstantin Pogorelov, Pål Halvorsen, and Michael Riegler. 2018. Mimir: An Automatic Reporting and Reasoning System for Deep Learning based Analysis in the Medical Domain. In *Proceedings of 9th ACM Multimedia Systems Conference, Amsterdam, Netherlands, June 12–15, 2018 (MMSys'18),* 6 pages.
https://doi.org/10.1145/3204949.3208129

## 1 INTRODUCTION

Machine learning has the potential in becoming an important tool in assisting medical professionals to perform medical diagnosis and giving aid in the administrative work that follows. Deep learning has already been shown to work well in various medical fields such as screening for skin cancer, where Esteva et al. [5] (in 2017) presented a deep convolutional neural network (CNN) with the ability to diagnose skin cancer at the level of a trained dermatologist. This shows that deep learning can successfully be applied to fields of medical expertise, but experts are still left with the work of documenting the procedure through written reports. With the amount of data gathered through medical examinations rapidly increasing, we need a way to process this information without drowning clinicians in administrative work.

One solution to this problem is through automatic methods, e.g. deep learning, where the collected data is automatically compiled into summaries, conveying key aspects from the medical procedure. This would not only relieve doctors from parts of the administrative process, but could be used as a teaching tool for medical students. Through multimedia enriched reports, medical doctors in training can learn based on real data according to case-based teaching and problem-based learning strategies. Thus, multimedia summarization for automated report generation is a much needed feature [25].

A major obstacle with using complex automatic methods is that the inner workings are often hard to understand, making it difficult to determine how and why it produces its results, i.e., deep learning

is often used as a "black box". This is especially problematic in the field of medicine among others, where the doctors need to justify a decision besides referring to the system itself. To the best of our knowledge, this is yet an unexplored area of research. Moreover, no open-source software exists that could support researchers in both domains, computer science and medicine, to perform much needed research in this direction.

In an effort to open this "black box" and assist in the documentation of medical examinations, we present *Mimir*, an automated multimedia reporting system, which goes beyond the creation of medical reports by adding a level of understanding and explainability through methods of looking into a deep neural networks decision process. The presented open-source software can be used for easy retrieval and reuse of data for automatic report generation, comparisons, teaching and research. As a first use-case, we use live colonoscopy, which is the conventional (and gold standard) method of screening the large intestine. Through insertion of a long flexible tube equipped with a tiny camera into the anus, it allows for direct inspection of the bowel mucosa. This plays an essential role in the diagnosis of various abnormalities commonly found in the lower gastrointestinal (GI) tract, such as inflammation, colorectal cancer and its precursors (polyps). Before starting *Mimir*, we developed a live detection system [26, 27] which analyses a direct video stream from a colonoscopy, and gives live visual feedback whether or not anything is detected [28]. This however, does not explain why the system signaled a detection and does not provide any form of text summaries of the overall examination process.

We aim not to just create reports containing text and most representative multimedia content such as images or videos, but also to explain to the users why a certain image has been identified as relevant. The main contribution therefore is to provide researchers and domain experts a novel way of using intermediate visual representations of deep neural network layers and results to increase understanding, trust and usefulness. The representations created by the system can be used for example in disease detection scenarios.

Below, we briefly describe the system based on Google's TensorFlow, give a brief introduction to the code and installation, and discuss some examples of how to use *Mimir*.

## 2 RELATED WORK

Over the last few years, deep learning has proved to be a powerful tool in many fields and is now (2018) considered the gold standard in many areas such as language translation, object recognition and image captioning [17]. However, generation of quality medical reports goes beyond transforming explicit information from one media to another. It often involves multiple different forms of media, which must be combined in order to argue and justify the diagnosis of a medical expert.

The current practice of reporting medical procedures is an essential, yet cumbersome, part of a clinicians' daily work. Research shows that approximately one-sixth of U.S. physicians working time is spent on administrative tasks, taking time away from direct-patient care and lessening job satisfaction [35].

In addition, within GI endoscopy, there is a general lack of language standardization, which may result in poor communication between health care providers. Thus, following a systematic approach to document the findings of an endoscopic procedure would be favorable in an attempt to achieve a certain level of consistency within GI reporting. An automated reporting system based on automatic video analysis would be extremely helpful in this regard, and help contribute to the implementation of the *Minimal Standard Terminology* (MST) recommended by the World Endoscopy Organization (WEO). Additionally, the standardization of medical reporting related to endoscopic procedures is listed as a requirement by the European Society of Gastrointestinal Endoscopy (ESGE) [4].

In the field of medicine, data driven methods can be questionable if the results are not reproducible or comprehensible by the medical experts using them. With deep learning in particular, the results of automatic recognition are extremely helpful, but we are still unable to fully understand the rationale behind the decisions made by the algorithm. This has lead to a trade-off between more comprehensible models and models that yield a higher accuracy, where simpler models are often chosen over those with higher accuracy as they are typically easier to interpret. Recent developments have provided theoretical and visual approaches to better understand the decisions made by a deep neural network. Theoretical approaches rely on describing the underlying mathematics, taking a closer look at how the individual mathematical properties produce a given result [18, 34]. This is useful, but interpreting such descriptions require a deep understanding of the math and technology of deep learning, something we cannot expect end-users to have. Visual approaches try to present layers using a variety of visualization techniques such as saliency maps or other forms of visual representations (texture maps, heat maps, etc.) [29, 37] and come closer to gaining a better understanding of the classification process without detailed technical knowledge of the underlying system.

It is worth noting that medical doctors indicated that a tool for automatic text generation was not that important to them. It was more important for them to understand the underlying analysis process, and receive support in generating high quality documents through consistent means [25]. *Mimir* aims to meet them halfway. By including the doctors in the analysis process, we give them an intuitive way to understand how and why the system produces its results.

In sum, the goal is to create a tool that aids in the production of a structured and semantically correct reports, composed of text and images taken from a medical procedure (GI endoscopy in our case). Moreover, the tool needs to make the process understandable and reproducible for non-technical users to ensure the trust of the doctors and patients involved.

A recent approach [15] investigates the possibility of creating reports from x-ray images employing neural image captioning methods [36]. A network is trained from a dataset of images along with the reports. Closest to our approach is the work described in [38], where microscope images are fed to a neural network to generate reports and retrieve relevant images of symptoms in addition to visualization of the attention of the network to support the rationale of the decision made by the network. Both approaches focus on images already classified as relevant by being part of a diagnostic process, whereas the second paper adds the dimension of the rationale of the generated report.

## 3 SYSTEM DESCRIPTION

*Mimir* can be described as a framework with three main functionalities:

- The system was designed to aid medical doctors in making informed decisions regarding diagnosis of diseases found during examinations, such as diagnosis of disease found in the GI tract during a colonoscopy.
- *Mimir* creates automatic reports based on the automatic analysis of images and videos and reduces the time spent on the administrative tasks that follow an endoscopic examination, e.g., documentation by written reports. This is shown in figure 1 where the doctor uses the system to understand the analysis done by the neural network, and use this information to reach a diagnosis and generate the accompanied report.
- *Mimir* can be used by researchers and engineers designing deep learning architectures such as CNNs to gain a better understanding of the evaluation and reactions of their models, e.g., by understanding which parts of an image confuse the algorithm and if additional pre-processing steps are needed.

In *Mimir*, we use a deep CNN to analyse image or video data to perform different classification task, e.g., automatic detection of diseases. This process is made transparent to the users through a tool that dissects the individual layers of a CNN, making it possible to see the basis for the decisions made by the system and on what regions of an image the algorithm activates for a target class. This is a critical piece in building trust among users of the system like medical professionals who need to rely on the systems output without the technical knowledge of the internal workings of a CNN. Additionally, it allows for discovering fallacies within the model itself and the dataset used to train it.

Using the guided grad-CAM technique [30], we generate visual representations of an image as it moves through the network, showing what regions of the image correspond to a target class at the point of a selected layer. The process is shown in figure 4, and starts with the user selecting an input image, target layer and target class using the web-interface (Shown at the bottom of figure 4). Based on the selection, the system generates three visualizations of the image (the three visualizations are shown in figure 2 together with the original image). Figure 2a is the original image before any processing. Figure 2b is a grad-CAM (A generalization of class activation map (CAM) [39]) representation of the image which shows what regions of the image correspond the the selected target class. Figure 2c is the saliency map generated using guided back-propagation, this shows the positive activations of the target layer, and is not class specific. Figure 2d depicts the guided grad-CAM representation of the image, which is a combination of the grad-CAM and saliency map. From the three visualizations, the system presents the grad-CAM and the guided grad-CAM to the user.

Figure 3 shows five guided grad-CAM representations of an image containing a polyp using polyp as the target class, each corresponding to the last convolutional layer in the five convolutional blocks of a VGG-19 CNN. The written reports are produced through a "what you see is what you get" (WYSIWYG) editor, with additional options for image attachments.

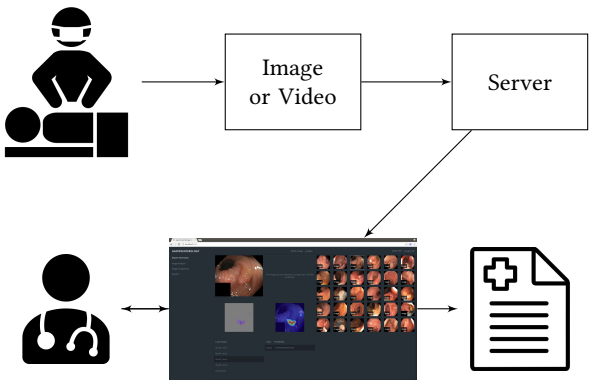
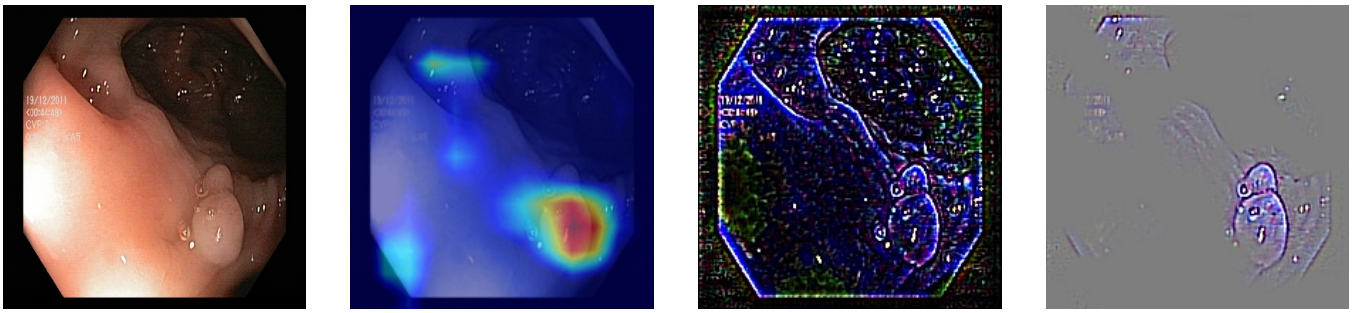



**Figure 1: Reporting system workflow. Images and videos are collected and analyzed during the examination. After the examination, the system can give intermediate insights from the neural network for the presented findings and a modifiable report draft is presented in order to produce a final report including text, representative images and video clips.**

## 4 CODE

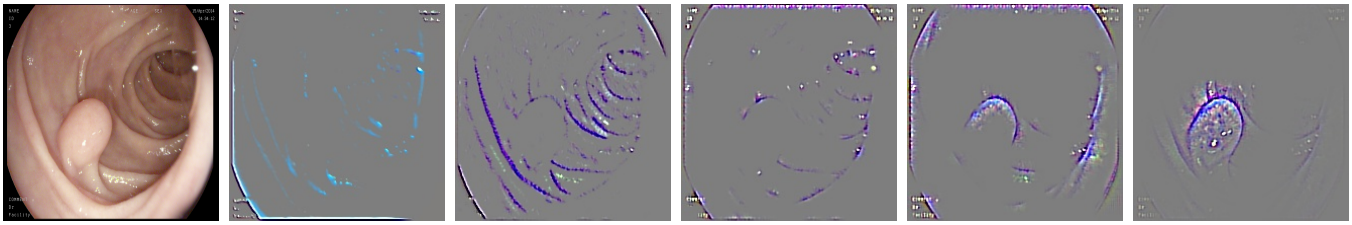
The code repository [14] contains the server and web application, clearly separated in their respective directories. The web application uses a standard flux architecture [6], implemented using React [8] and Redux [1]. The code is fully documented and tested using the testing framework Jest [7]. The advantage of making a client web application is the ease of access and portability of being available on any device that supports a web browser. The server is written in Python (using the micro-framework Flask[10]), and is accessible through a RESTful [24] API, with endpoints for interaction with the underlying deep neural network. As mentioned previously, the image/frame analysis is done using deep learning, specifically a deep CNN. The CNN uses a standard VGG-19 architecture [31] trained on the Kvasir version 2 dataset [22] and is implemented using the Keras deep learning framework [16] with a Tensorflow backend [2]. *Mimir* is licensed under the terms of the GNU General Public License (GPL) version 3, as published by the Free Software Foundation and available on Github [14].

The image/frame visualizations are done using the guided grad-CAM approach [30], and is generated on the fly once the user selects an image, target layer and target class to visualize. A target layer and target class can be selected individually from their respective lists (as seen in the web-interface of figure 4). The layer selection list contains each convolutional layer in the underlying CNN, and the class selection list contains each class supported by the system. The guided grad-CAM technique combines the class discriminative properties of a CAM with pixel level detail of guided back-propagation saliency maps [32]. Our implementation is based on the Selvaraju et al. paper [30] and is implemented using Keras backend functions. The current system uses two image representations to explain the CNN, grad-CAM and guided grad-CAM. The overall process of generating these two visualizations can be broken into three parts;



(a) Original Image (b) Grad-CAM (c) Guided Backprop (d) Guided Grad-CAM

**Figure 2: Image representations used by the reporting system to explain decisions.**



(a) Original (b) Block 1 (c) Block 2 (d) Block 3 (e) Block 4 (f) Block 5

**Figure 3: Guided grad-CAM representation of an image at the last convolutional layer of each convolutional block.**

(1) Generate a grad-CAM representation using the a target layer and target class with respect to the input image.
(2) Generate a guided back-propagation saliency map using the same target layer as used when generating the grad-CAM with respect to the input image.
(3) Combine the grad-CAM with the saliency map made in the previous two steps to produce the guided grad-CAM visualization.

Evidently, the grad-CAM is an intermediate step of the guided grad-CAM process, so both representations are created through the same process. A visual representation of the process can be seen in figure 4.

The visualization process starts once the user has selected an image, layer and class for further inspection. With an image, target layer and target class selected, we calculate the gradient of the target layer using the loss of the target class in regards to the image. These gradients are globally average pooled to get the weights, which is multiplied with output of the target layer and passed through a relu function to produce the grad-CAM. The grad-CAM is re-sized back to the dimensions of the original image and its values squashed between 0 and 1 before a blue-red heat map filter is applied.

In order to generate the guided back-propagation saliency map, we start by replacing the activations of the original network with a slightly modified relu function. During back-propagation, a traditional relu would let all gradients whose inputs where larger than 0 pass. We change this by adding an additional rule which discards all gradients that have value below 0 (i.e. negative gradients), thereby only back-propagating the positive influence on the activations. With this modified network we calculate the gradients of the target layer with respect to the input image, these gradients represent our saliency map.

Once the grad-CAM (Figure 2b) and saliency map (Figure 2c) have been computed, we simply multiply them together to produce the guided grad-CAM (Figure 2d) representation. This together with the grad-CAM is used in our system.

## 5 INSTALLATION

As mentioned in section 4, the system is built using the micro-framework Flask, which includes a built-in development sever, making it easy to start a local instance of the application. Note that the development server is not meant to be deployed to a production environment. For a production environment we recommend deployment using a popular web server such as Nginx [19] or Apache HTTP Server [3], or by using the pre-built Docker Image available through Docker hub [13]. There are two primary ways of getting the system up and running, pulling the git repository from Github[1] [14] or pulling the pre-configured docker image from Docker hub[2] [13].

Setting up the system using the git repository requires multiple steps of pre-configuration before we can launch the local development server. This includes;

(1) Setup a Python 3.6 run-time environment with the necessary dependencies.
(2) Configure Keras (2.0.8) [16] to use Tensorflow [11] as a backend.
(3) Install OpenCV [33] with FFmpeg [9] support.

[1] https://github.com/acmmmsys/2018-Mimir/
[2] https://hub.docker.com/r/stevenah/mimir/

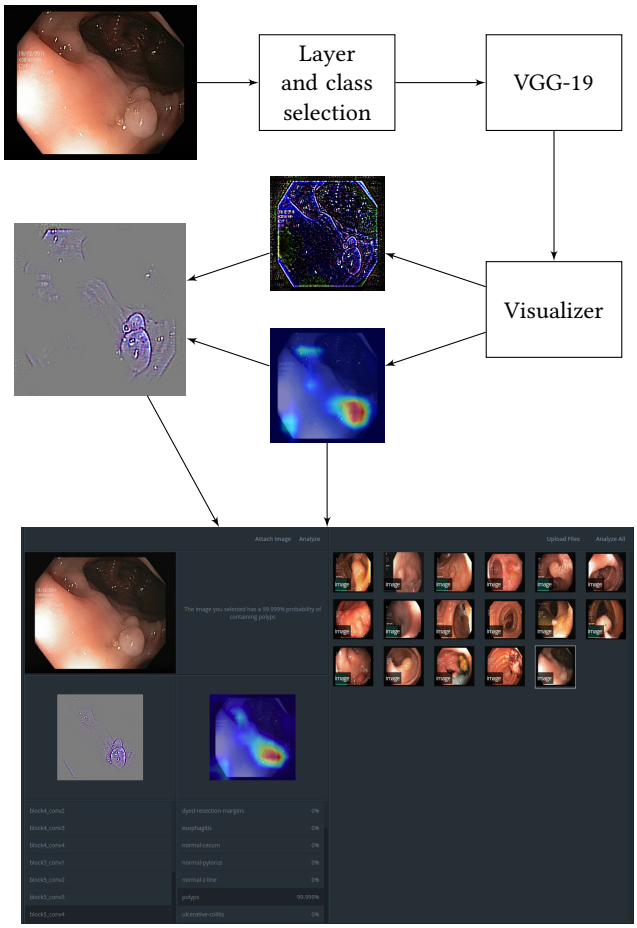



**Figure 4: An overview of how we produce the two visualizations included in the image analysis, and how it is presented in the user interface where a visualization of the different convolutional blocks can be selected.**

(4) Install CuDNN [21] and CUDA toolkit [20] for GPU support (this step is optional, but highly recommended).

A more detailed setup and configuration guide can be found in the applications Github repository. With the environment setup, we can launch a local development server by running `app.py` using Python 3.

For an easier setup, we recommend using the pre-built Docker image available at Docker hub [13]. The container includes a pre-configured Python environment with all the necessary dependencies installed, CUDA 8 and cuDNN 6 for Nvidia GPU support, and served using an Ngnix server instance.

## 6 USAGE

The web application can be accessed using the configured host IP and port. The following examples will describe typical scenarios we imagine this tool being used for.


**Figure 5: An example of an automatic generated report. The red area marked (1) shows the editable text fields. The green area (2) shows the images chosen for the report. Report based on sample taken from Wrestling the Octopus [12].**

### 6.1 Example Scenario A - Verify the Prediction of a Diagnosis

After getting the diagnosis based on the analysis of the colonoscopy examination video, we would like to verify that the network does in fact detect the diagnosed abnormality presented. After the examination, the frames where abnormalities are detected are automatically presented to the user on the image analysis web-page. For a given frame, the user can look through the network and verify that the network does in fact detect the abnormality related to the diagnosis. An example is shown in figure 2 where we clearly see that the network detects the polyp located in the lower right corner of the image. Note that not all detections are this obvious, and the additional image representations are thus even more useful when abnormalities are difficult to detect.

### 6.2 Example Scenario B - Generating a Colonoscopy Report

After a colonoscopy, the video produced is automatically passed through the system and analysed for abnormalities. Based on the diagnosis, the system would present images that support the diagnoses (which can be further examined as described above in

section 6.1). This would save the user time by not having to screen the frames of the video for the diagnosed abnormality and manually select image candidates.

The report generation tool provides basic text editing through a WYSIWYG interface, with additional options for adding images to support the findings described in the report. The tool presents a live preview of the printed document, which may be modified by clicking the various text fields of the report. Images may be manually or automatically added through image uploads or by taking images already part of the system. Note that the current format of the report is taken from Wrestling the Octopus [12], and is used just as an example. In a real world use-case, the format of the report would be tailored to the needs of the institution. An example report can be found in Figure 5 with text and pre-selected images (that the user can change).

## 7 CONCLUSION

Nowadays, neural networks are widely used, but there is still a lack of understanding when it comes to how they operate and on what their output is based on, even more so among non-technical users. This is may be sufficient for many fields, but in mission-critical areas such as medicine (among others), the clinicians often need to understand why a particular marking is detected. To improve the understanding of the internal decision process of a deep neural network, and to build trust among its users, we made the source code of our system publicly available. *Mimir* allows for dissecting of deep neural networks, enabling investigation and understanding of the networks layers and outputs. Our system can also use this information to create automatic reports from the analysis of images or videos. In this paper, we have briefly described the system based on Google's TensorFlow, given a introduction to the code and installation, and discussed some examples of how to use *Mimir*.

## REFERENCES


[1] 2018. Redux. (2018). https://redux.js.org/

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, and others. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).

[3] Apache. 2018. Apache HTTP Server Project. (2018). https://httpd.apache.org/

[4] Michael Bretthauer, Lars Aabakken, Evelien Dekker, Michal F Kaminski, Thomas Rösch, Rolf Hultcrantz, Stepan Suchanek, Rodrigo Jover, Ernst J Kuipers, Raf Bisschops, and others. 2016. Reporting systems in gastrointestinal endoscopy: Requirements and standards facilitating quality improvement: European Society of Gastrointestinal Endoscopy position statement. *United European gastroenterology journal* 4, 2 (2016), 172–176.

[5] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* 542, 7639 (feb 2017), 115–118. https://doi.org/10.1038/nature21056

[6] Facebook. 2018. Flux. (2018). https://facebook.github.io/flux/

[7] Facebook. 2018. Jest. (2018). https://facebook.github.io/jest/

[8] Facebook. 2018. React. (2018). https://reactjs.org/

[9] FFmpeg. 2018. FFmpeg. (2018). https://www.ffmpeg.org/

[10] Flask. 2018. Flask. (2018). http://flask.pocoo.org/

[11] Google. 2018. Tensorflow. (2018). https://www.tensorflow.org/

[12] Nigel H. 2015. The Crohnoid Blog. (2015). http://www.wrestlingtheoctopus.com/the-a-to-z-of-my-crohns/

[13] Steven Hicks. 2018. Mimir Docker Repository. (2018). https://hub.docker.com/r/stevenah/mimir/

[14] Steven Hicks. 2018. Mimir Github Repository. (2018). https://github.com/acmmmsys/2018-Mimir

[15] Baoyu Jing, Pengtao Xie, and Eric Xing. 2017. On the Automatic Generation of Medical Imaging Reports. *arXiv preprint arXiv:1711.08195* (2017).

[16] Keras. 2018. Keras: The Python Deep Learning library. (2018). https://keras.io/

[17] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[18] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2017. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* (2017).

[19] NGINX. 2018. NGINX. (2018). https://nginx.org/en/

[20] Nvidia. 2018. Nvidia CUDA Toolkit. (2018). https://developer.nvidia.com/cuda-toolkit

[21] Nvidia. 2018. Nvidia CuDNN. (2018). https://developer.nvidia.com/cudnn

[22] Konstantin Pogorelov, Kristin Ranheim Randel, Carsten Griwodz, Sigrun Losada Eskeland, Thomas de Lange, Dag Johansen, Concetto Spampinato, Duc-Tien Dang-Nguyen, Mathias Lux, Peter Thelin Schmidt, Michael Riegler, and Pål Halvorsen. 2017. KVASIR: A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection. In *Proc. of MMSYS*. 164–169. https://doi.org/10.1145/3083187.3083212

[23] Konstantin Pogorelov, Michael Riegler, Sigrun Losada Eskeland, Thomas de Lange, Dag Johansen, Carsten Griwodz, Peter Thelin Schmidt, and Pål Halvorsen. 2017. Efficient disease detection in gastrointestinal videos – global features versus neural networks. *Multimedia Tools and Applications* 76, 21 (01 Nov 2017), 22493–22525. https://doi.org/10.1007/s11042-017-4989-y

[24] Leonard Richardson and Sam Ruby. 2007. *Restful Web Services* (first ed.). O'Reilly.

[25] Michael Riegler, Mathias Lux, Carsten Griwodz, Concetto Spampinato, Thomas de Lange, Sigrun L Eskeland, Konstantin Pogorelov, Wallapak Tavanapong, Peter T Schmidt, Cathal Gurrin, and others. 2016. Multimedia and Medicine: Teammates for Better Disease Detection and Survival. In *Proc. of ACM MM*. 968–977.

[26] Michael Riegler, Konstantin Pogorelov, Pål Halvorsen, Thomas de Lange, Carsten Griwodz, Peter Thelin Schmidt, Sigrun L. Eskeland, and Dag Johansen. 2016. EIR - Efficient Computer Aided Diagnosis Framework for Gastrointestinal Endoscopies. In *Proc. of CBMI*.

[27] Michael Riegler, Konstantin Pogorelov, Jonas Markussen, Mathias Lux, Håkon Kvale Stensland, Thomas de Lange, Carsten Griwodz, Pål Halvorsen, Dag Johansen, Peter T Schmidt, and Sigrun L. Eskeland. 2016. Computer Aided Disease Detection System for Gastrointestinal Examinations. In *Proc. of MMSys*.

[28] Michael Riegler, Konstantin Pogorelov, Jonas Markussen, Mathias Lux, Håkon Kvale Stensland, Thomas de Lange, Carsten Griwodz, Pål Halvorsen, Dag Johansen, Peter T. Schmidt, and Sigrun L. Eskeland. 2016. Computer Aided Disease Detection System for Gastrointestinal Examinations. In *Proc. of MMSYS*. 29:1–29:4. https://doi.org/10.1145/2910017.2910629

[29] Christin Seifert, Aisha Aamir, Aparna Balagopalan, Dhruv Jain, Abhinav Sharma, Sebastian Grottel, and Stefan Gumhold. 2017. Visualizations of Deep Neural Networks in Computer Vision: A Survey. In *Transparent Data Mining for Big and Small Data*. Springer, 123–144.

[30] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. 2016. Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization. *CoRR* abs/1610.02391 (2016). arXiv:1610.02391 http://arxiv.org/abs/1610.02391

[31] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014). arXiv:1409.1556 http://arxiv.org/abs/1409.1556

[32] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. 2014. Striving for Simplicity: The All Convolutional Net. *CoRR* abs/1412.6806 (2014). arXiv:1412.6806 http://arxiv.org/abs/1412.6806

[33] OpenCV team. 2018. Open Source Computer Vision Library (OpenCV). (2018). https://opencv.org/

[34] Rene Vidal, Joan Bruna, Raja Giryes, and Stefano Soatto. 2017. Mathematics of Deep Learning. *arXiv preprint arXiv:1712.04741* (2017).

[35] Steffie Woolhandler and David U Himmelstein. 2014. Administrative Work Consumes One-Sixth of U.S. Physicians' Working Hours and Lowers Their Career Satisfaction. 44 (10 2014), 635–42.

[36] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proc. of ML*. 2048–2057.

[37] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. 2015. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579* (2015).

[38] Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. 2017. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *Proc. of IEEE CVPR*. 6428–6436.

[39] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. 2015. Learning Deep Features for Discriminative Localization. *CoRR* abs/1512.04150 (2015). arXiv:1512.04150 http://arxiv.org/abs/1512.04150