Integrating Learning and Reasoning Services for Explainable Information Fusion

Dan Harborne*, Chris Willis†, Richard Tomsett‡, Alun Preece*
*Crime and Security Research Institute, Cardiff University, Cardiff, UK; contact email: HarborneD@cardiff.ac.uk
†BAE Systems AI Laboratories, Chelmsford, UK
‡IBM Emerging Technology, Hursley, UK

***Abstract*—We present a distributed information fusion system able to integrate heterogeneous information processing services based on machine learning and reasoning approaches. We focus on higher (semantic) levels of information fusion, and highlight the requirement for the component services, and the system as a whole, to generate explanations of its outputs. Using a case study approach in the domain of traffic monitoring, we introduce component services based on (i) deep neural network approaches and (ii) heuristic-based reasoning. We examine methods for explanation generation in each case, including both transparency (e.g, saliency maps, reasoning traces) and post-hoc methods (e.g, explanation in terms of similar examples, identification of relevant semantic objects). We consider trade-offs in terms of the classification performance of the services and the kinds of available explanations, and show how service integration offers more robust performance and explainability.**

***Index Terms*—information fusion, explainability, interpretability, machine learning, reasoning, distributed systems**

I. Introduction

Our goal is to facilitate situational understanding by human analysts by providing an open, distributed information fusion architecture that integrates machine learning and reasoning services operating on multimodal data feeds. We broadly subscribe to the view that an integrated learning and reasoning system can be viewed as a dynamic composition of simpler models such that the composed system is able to answer questions that the individual models cannot [1]. In our case, we require that the system not only provide human analysts with classified or inferred assessments of a situation, but also be able to provide explanations for its assessments in terms of a typology of model interpretability [2]. Moreover, we aim to take advantage of the compositional architecture by exploiting semantic relationships between model outputs to improve generated explanations, especially for information fusion processes involving sub-symbolic models where interpretability remains a challenging problem, including deep neural networks, DNNs [3].

We require that the architecture be distributed to allow for information processing (including pattern recognition and inference) to occur anywhere on the network, and also to allow sharing of heterogeneous services from multiple providers. Sharing brings the additional requirement that there must be a degree of control over service access and information flow, e.g., for privacy or security reasons.

The work presented in this paper is framed as a case study, in which a representative situational understanding problem and available multimodal data feeds facilitate experiments in:

1) integrating machine learning and reasoning models via a lightweight service-oriented architecture;
2) exploring trade-offs between (i) model performance for situational assessment and (ii) interpretability for explanation generation; and
3) enabling the imposition of constraints to control information flow.

The paper is organised as follows: Section II summarises prior work in information fusion and explainability; Section III introduces the application used as the basis for our case study; Section IV describes the main services: a machine learning service based on a convolutional neural network (CNN) and a composition of services that comprise a reasoner; Section V describes the explanations generatable for each service; Section VI concludes, pointing to future work.

II. Related Work

Our work aims to support situational understanding via information fusion, with a focus on higher fusion levels and human-in-the-loop processes [4]. While much work in fusion of multimodal data addresses signal processing techniques applied at lower (data) levels (e.g., [5]) our interest in integrating heterogeneous machine learning and reasoning services leads us to consider late fusion at higher (information) levels (e.g., [6]). Moreover, we require services to provide explanations for their outputs and, where outputs are fused, we require the fusion of those explanations also.

Explainability in artificial intelligence systems has been recognised as a problem for several decades, and was extensively studied in the 1980s and 1990s in the context of symbolic reasoning ('expert') systems (e.g., [7], [8]). The focus then was on effective means to make the reasoning of knowledge-based systems *transparent* to end-users. Fundamentally, the approaches sought to frame explanations in terms of *reasoning traces* (e.g., chains of rule firings or proof trees) and *component data* (i.e., input data that triggered the rules or grounded the proof). While non-trivial, this work benefited from these systems being symbolic rather than sub-symbolic; their internal elements were largely explicable.

The resurgence of interest in sub-symbolic approaches in recent years, chiefly DNNs, has led renewed interest and

concerns regarding explainability[1]. System *transparency* remains a key issue, but it is also recognised that 'traces' in terms of DNN weights are not explicable and, often, *post-hoc* explanations are more useful [2]. The dominant approach to addressing transparency in DNNs is in image classification systems, to associate an output class with the parts of an input image that had the greatest weight in determining the classification; e.g., *saliency mapping* identifies regions of similarly-weighted pixels in an input image that contribute positive weight towards a particular output class [9], [10].

Common post-hoc explanation approaches for DNNs include *explanation-by-example* and *text explanation*. The former draws on earlier work in case-based reasoning wherein an explanation is framed in terms of a selection of labelled cases computed to be similar to the input case [11], [12], [13]. The latter employs object detection techniques to identify meaningful sub-elements of the input and constructs an explanation based on these elements, in a manner similar to automatic image captioning [14].

III. Testbed Application

Our testbed was chosen as an exemplar application where open data and pre-existing services were readily available, and where machine learning and reasoning services could plausibly operate at a range of semantic scales (low to high-level). We selected the problem of monitoring and predicting traffic congestion. In many cities, multiple organisations offer open sources of information. For example, Transport for London (TfL) offers an application programming interface (API) to view imagery and video from their traffic cameras placed around London[2], while Open Street Maps (OSM) offers information about roads, e.g., speed limit[3].

A. Testbed System Architecture and Design

The structure of the system is shown in Figure 1. The figure shows information flows from data sources through processing services to decision-support classifications. Dark grey arrows show the path information takes to produce congestion classifications (ratings) and light grey arrows show information flows that produce explanations of those ratings. There are two service chains producing classifications: one via a CNN (only) and a second via a more complex composition of services that feed in to a reasoning service. We refer to the CNN as the *congestion classifier* and the latter as the *congestion reasoner*. Section IV details these services and Section V examines the kinds of explanations that can be generated from each.

B. Data Collection and Labelling

TfL provides still images and video sequences (of a few seconds duration) from over 1,200 cameras, released at a 5-minute refresh rate. This paper focuses principally on the still images; these are 352×288 pixels. We collected images and video sequences from 691 cameras over a period of 23 days, including imagery at all times of the day and night, in different location types, and over a range of traffic and environmental conditions. See Figure 2 for example images illustrating some of the variations in lighting, traffic and road configuration present in the dataset.

A subset of the imagery covering five locations over a 24 hour period was selected for ground truth labelling. For this, a web-based image annotation tool[4] was developed to allow users to label each image as one of: 'congested', 'uncongested', 'unknown' or 'broken'. Users were guided to use the first two labels for images where they were confident that the traffic was congested or not, to use 'unknown' where they were uncertain, and 'broken' to label images that were blank or otherwise unreadable (an artifact of the TfL system). Use of this tool by 12 users yielded a dataset consisting of 4,117 images labelled by at least one annotator. Once the broken images were rejected a total of 3,967 usable labelled examples remained. Taking the most prevalent labelling for each image revealed that 57% were uncongested and 43% congested. In order to take account of the non-binary nature of the phenomena of interest, and differing opinions expressed

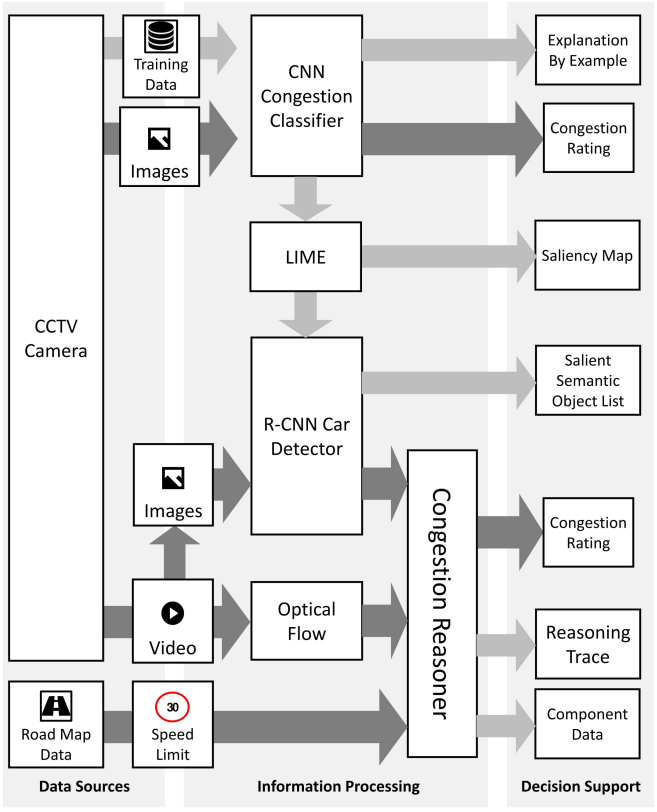

Fig. 1. Testbed system architecture: dark grey arrows show information flow leading to congestion ratings; light grey arrows show information flow to generate explanations for ratings

[1]Recent machine learning literature favours the term 'interpretability' over 'explainability'. We use the terms interchangeably here.

[2]http://www.trafficdelays.co.uk/london-traffic-cameras/

[3]http://www.openstreetmap.org/

[4]https://image-annotation.eu-gb.mybluemix.net/

Fig. 2. Example Images: uncongested (left), borderline (middle) and congested (right)

when labelling the imagery, soft class memberships were derived for use in training and validation. These took into account the labels supporting a class, against the class and for the 'unknown' labels:

$$p_i = \frac{v_i + u + 1}{\sum_{j=1}^{N}(v_j + u + 1)}.$$

Here, v_i are the number of annotations assigned to class i and u are the number of annotations for the unknown class. This soft class membership avoids issues caused by the hard classification of borderline cases adversely biasing results. For classifier development the images were randomly partitioned into a training set of 80% of images and 10% for each of the validation and test sets. (This approach resulted in acceptably good performance for our purposes with relatively fast training; for a production system we would perform cross-validation.)

The services are integrated using the Node-RED programming environment for Internet of Things applications (nodered.org). System code and data is available at: https://osf.io/wm3t9

IV. Machine Learning and Reasoning Services

A. CNN Congestion Classifier

The congestion classifier uses a pair of DNNs. The first uses the GoogLeNet [15] Inception network, pre-trained on ImageNet data, for feature extraction. A feature vector for transfer learning is tapped-off before the network's fully connected layer. The feature vector is passed into a five-layer fully-connected network to transform image features into a classification assessment. This output (congestion rating) is the congestion class conditional probability. Based on previous experience, we used hyperbolic-tangent and softmax activation functions and trained the network using the Adam optimizer until the cross-entropy on the validation set stops improving. Both networks are implemented in TensorFlow [16].

Training Method: The training, validation and test datasets were generated by collecting a number of 200×200 pixel sub-images from each starting image. Firstly, the central sub-image was collected, from which we produced eight overlapping images formed by offsetting the selection region by 25 pixels to the left, right, up and down. This was repeated for left-right flipped variants yielding an augmented dataset of over 57,000 training and 7,000 validation and test images respectively. The sub-images generally omit the TfL annotations at the edges of the originals (Figure 2).

Results: A selection of classified images are shown in Figure 2: an image classified as uncongested on the left, a borderline case in the centre and an image classified a congested on the right. Performance on the test set as a whole (Table I) indicated that the proportion of images classified according to most prevalent label was over 95%, and examination of the miss-classified images revealed that most were on the classification borderline (with probability close to 50%).The recorded precision and recall are 0.98 and 0.96 respectively.

		Predicted class		
		Congested	Not Congested	Total
True class	Congested	1616	66	1682
	Not Congested	35	2269	2304
	Total	1651	2335	

TABLE I
CONFUSION MATRIX FOR CNN CONGESTION CLASSIFIER (TFL IMAGERY FROM 5 LOCATIONS OVER A 24 HOUR PERIOD)

B. Congestion Reasoner

As shown in Figure 1, the congestion reasoner involves an integration of multiple services and data sources and was intentionally crafted as a lightweight heuristic-based service.

Development Method: Video clips from the TfL API are passed to an optical flow (OF) algorithm[5] which generates information in the form of blobs in motion. Semantically, these detected entities cannot be assigned any meaning and we are unsure as to whether the blobs are cars, people or something else. Therefore, to make this information more usable, key frames of the video are sent to an object detector (a retrained instance of the VGG-16 regional-convolutional neural network, R-CNN, model [17]) able to identify the cars in the frames. We fuse the information regarding the detected cars together with the detected blobs (including their pixel velocities across the video) which results in a fused output of detected cars and their pixel velocities.

[5]Mixture of Gaussian (MOG2) & Lucas-Kanade (github.com/itseez/opencv)

Only using the R-CNN on key frames (specifically every 30 frames) reduces the computational cost of the service chain (given that the OF algorithm is comparatively inexpensive when compared to the R-CNN). This approach also demonstrates how services can be used within a service composition whilst adhering to possible usage constraints (such as limiting the number of calls that can be made to a service).

Once the pixel velocities of the cars for a video are obtained, they can be averaged and this can be compared to the speed limit of the location. From the ratio between the velocity and speed limit, a conclusion can be drawn about the value range that the ratio commonly moves within. From this we can derive a heuristic that determines at what point the ratio indicates low and high levels of traffic flow, allowing us to infer the level of congestion.[6]

Results: For the generation of results for this service chain, ground truth for TfL video was required. To produce this, we utilised the labels assigned during the CNN image data set generation. These images were originally produced by pulling frames from the TfL videos from across a single day. As such, the labels assigned to these frames could be aggregated per video to provide a ground truth label for the source video and these labeled videos (733 in total) could form the data set for testing. The performance of the reasoner on the test set is shown in Table II. The recorded precision and recall are 0.79 and 0.87 respectively.

		Predicted class		
		Congested	Not Congested	Total
True class	Congested	283	51	334
	Not Congested	91	308	399
	Total	374	359	

TABLE II
CONFUSION MATRIX FOR CONGESTION REASONER CLASSIFYING CONGESTION IN TFL VIDEO (5 LOCATIONS ACROSS 24 HOUR PERIOD)

V. EXPLANATION GENERATION

In this section, we outline explanation generation techniques applicable to our two congestion detection methods, together with additional services used to support them; Table III summarises these.

A. CNN Congestion Classifier Explanations

Transparent explanation via saliency map: Transparent explanations are those which are generated using the signals that pass through the DNN model and thus aim to directly indicate the reasons for model output. For this work we used the LIME [9] tool to generate a saliency map in the form of highlighted regions of the original input. These region were important in the model's assessment of the likelihood of the output label (strong evidence supporting the label or strong evidence against it). Figure 3 shows an input image and its corresponding saliency map derived using LIME.

[6]Our heuristic function for computing the traffic flow ratio f where s is the location speed limit is: $f = tanh(\bar{v} \times tanh(s \times 0.08))$. If $f \leq 0.4$ we infer congestion. The constants 0.08 and 0.4 were empirically derived from observations of traffic behaviour at a range of locations.

Explanation		Detection technique		Supporting services	
		CNN	Reasoner	LIME	R-CNN
Transparent	Saliency map	•		•	
	Component data		•		
	Reasoning trace		•		
Post-hoc	Semantic objects	•		•	•
	Training examples	•			

TABLE III
EXPLANATION AVAILABILITY AND SUPPORTING SERVICES FOR CLASSIFIER AND REASONER SERVICE CHAINS

Fig. 3. Original input image (left) and saliency map (right: red regions show evidence towards 'congested', green regions show evidence towards 'not congested')

'Map forms' of explanation are a common approach to adding explainability to black box models. LIME and similar techniques (such as deep Taylor decomposition [18]) that produce maps have important common characteristics that define how they can be used to provide interpretations. One key characteristic is that, in order for an explanation to be human-interpretable, it will often need to be presented in the modality of, and in relation to the original input. This can lead to challenges with sharing the explanation if the original input data cannot be presented for some reason (e.g., privacy). The next section shows how post-hoc explanations that do not reveal the input data can address this problem.

Post-hoc explanation via salient semantic object identification: Unlike transparent explanations, post-hoc explanations do not directly derive from the signals of a model but still can provide justification for, or further insight into, the model's output. In the technique described in this section, the post-hoc explanation is generated using a transparent explanation (saliency map) as input rather than using the original raw input data. This provides assurance that the post-hoc explanation is related to the decision process of the model.

The light grey arrows in Figure 1 show the passing of the saliency map from LIME to the R-CNN detector that produces a list of detected *salient semantic objects* (SSOs)

that are present within the highlighted regions of the salient map and which have a close semantic relationship to the target label ('congestion'). To detect the SSOs, the saliency map is used to mask the original input image leaving only the regions of the image that had the highest impact on the model's classification decision. The set of semantically-relevant objects is provided by a knowledge base (KB); in our current work this is a custom KB, but open KBs can be used instead (e.g., conceptnet.io) Given a set of semantically-relevant objects, an open situational understanding system could in principle automatically discover available services trained to detect the relevant objects. In our case study, we already had a service available for detecting cars in imagery (the R-CNN), created for use in the reasoner service chain.

This explanation technique generates a list of SSOs which is meaningful in absence of the input image. An example is shown in Figure 4. As mentioned above, this allows the original image to remain private (the explanation service can return the classification and the SSO list as explanation without returning the input image). Moreover, where network bandwidth is very low, transmitting the object list requires a fraction of the bandwidth required to transmit even a thumbnail image.

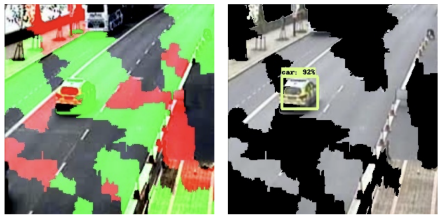

Fig. 4. This example shows a LIME saliency map (left), and an SSO identified in the salient region via the car detector service

The SSO list is also easily machine-processable, unlike the saliency map which is intended for direct interpretation by a human user. The SSO list can be used to verify whether the classifier has used features that are semantically relevant to the target class. The system can detect unusual cases (e.g., en empty SSO list) and automatically flag a classification as requiring the attention of a human, which can help direct user attention to cases where a human confirmation of a classification is necessary.

Post-hoc explanation via similar training set examples: A common post-hoc explanation technique involves retrieving similar previous cases, usually from the training set [2]. For this explanation to be available, the trained classifier must first be used to provide a rating for the original training set images (or a subset of them). When an explanation is required for a live classification, the system can look for similar classification ratings within the ratings of the training data and present examples alongside the current input image. This indicates what the model considers to be similar presence levels of the output class.

In our case study, for an image with congestion class conditional probability p, we display one example with a class conditional probability close to $\max(0, p - \epsilon)$ and one with a probability close to $\min(p + \epsilon, 1)$. We considered that this would provide a well-rounded impression of the model's internal representation. Figure 5 shows an example. Currently, $\epsilon = 0.1$; however, this can be configured as desired and further research could be conducted to explore the preferences of users.

Fig. 5. Explanation by training set examples: (left) training set example with congestion rating 0.1 less than input, (middle) input image, (right) training set example with congestion rating 0.1 more than input

B. Congestion Reasoner Explanations

Transparent explanation via component data: As the congestion reasoner uses a composition of services, the system is able to offer a transparent explanation in the form of the components of information that were used to make the final classification. Here, the speed limit, the traffic flow and the weighted ratio rating between the two are given to the decision agent in order to provide interpretability of the final conclusion generated by the service.

Transparent explanation via reasoning trace: As common in rule-based systems, the ability to easily explain inferences by providing a trace of the system's reasoning is possible. In our system, a rule is used that states if the ratio between the average car pixel velocity and the speed limit is lower than a threshold, the road is classified as congested. This rule can be presented to the decision maker to provide an explanation of how the service chain used the component data to infer the final classification.

C. Discussion

We compared the classifications produced by the congestion reasoner across the test set (733 videos) to the classifications produced by the CNN congestion classifier for imagery at matching location and times. We also compared both these sets of classifications to the respective counts of detected SSOs from the CNN classification. Table IV shows the correlation between (i) the congestion rating (congestion class conditional probability) provided by the CNN congestion classifier, (ii) the count of SSOs detected, and (iii) the congestion rating provided by the reasoner (i.e., $1 -$ traffic flow ratio). All correlations are significant at $p < 10^{-7}$. Notably, there is a strong correlation between the number of SSOs detected and the congestion rating of the CNN. This shows that the SSO list method does appear to offer a useful explanation for the CNN's classification.

These correlations give us confidence that the system shown in Figure 1 provides an integrated approach to traffic congestion detection and explanation. The CNN-based classifier

	SSO count	$1 -$ flow ratio
Congestion rating	0.61	0.41
SSO count	—	0.20

TABLE IV
SPEARMAN CORRELATION BETWEEN (I) CNN CONGESTION RATING, (II) COUNT OF DETECTED SSOS AND (III) $1 -$ TRAFFIC FLOW RATIO; ALL CORRELATIONS SIGNIFICANT AT $p < 10^{-7}$

can be considered as the primary congestion detection method due to its higher accuracy (Table I vs Table II). From this, we can generate statements in the form of 'This location appears congested/not congested' dependant on the classifier's output. Taking the congestion reasoner's output as a secondary method, we can add to this statement a clause that is either complementary ('This location appears congested and there appears to be a low level of traffic flow') or in disagreement ('This location appears congested but there appears to be a high level of traffic flow'). Moreover, the presence/absence of SSOs can help strengthen/weaken the confidence in the system's final classification and can indicate whether a human user needs to inspect the range of explanations offered.

VI. CONCLUSION AND FUTURE WORK

The main contribution of this paper is a systems approach to distributed information fusion integrating heterogeneous services, based on machine learning and reasoning approaches, with an ability to generate explanations of various kinds. Traffic congestion monitoring was chosen as an exemplar application featuring readily-available data of multiple modalities, plausible integration of machine reasoning and learning-based services, and semantic entities at lower and higher levels of abstraction. Through this case study, we have demonstrated how services based on (i) DNN approaches and (ii) heuristic-based reasoning can be combined in a common decision-support task. We examined the distinct kinds of explanation that can be generated for each type of service, including transparency (saliency maps for (i), reasoning traces for (ii)), and post-hoc methods (explanation in terms of similar examples or identification of relevant semantic objects for (i)). We explored trade-offs in terms of the classification performance of the services (in our case study, (i) was more accurate then (ii)) and the kinds of available explanations ((ii) was more transparent than (i)), and showed how service integration offers more robust performance and explainability by demonstrating that correlations between the service outputs can be exploited to generate richer explanations when services 'agree', and also when they 'disagree'.

Moving forward, we plan to expand the services used within the system to enable prediction of future traffic states, together with accompanying explanations for temporal predictions. We also aim to expand the range of data sources to include weather, textual traffic reports and events feeds, for use both in prediction and explanation.

ACKNOWLEDGMENT

This research was sponsored by the U.S. Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the UK Ministry of Defence or the UK Government. The U.S. and UK Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon. We thank Supriyo Chakraborty and the ICPRAI reviewers for helpful comments on drafts of this paper.

REFERENCES

[1] L. Bottou, "From machine learning to machine reasoning," *Machine Learning*, vol. 94, no. 2, pp. 133–149, 2014.
[2] Z. C. Lipton, "The mythos of model interpretability," in *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, 2017.
[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 5 2015.
[4] D. L. Hall and J. M. Jordan, *Human-Centered Information Fusion*. Artech House, 2010.
[5] D. Lahat, T. Adali, and C. Jutten, "Multimodal data fusion: An overview of methods, challenges, and prospects," *Proceedings of the IEEE*, vol. 103, no. 9, 2015.
[6] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, "Early versus late fusion in semantic video analysis," in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, 2005, pp. 399–402.
[7] B. Buchanan and E. Shortliffe, *Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, 1984.
[8] W. Swartout, C. Paris, and J. Moore, "Explanations in knowledge systems: design for explainable expert systems," *IEEE Expert*, vol. 6, no. 3, pp. 58–64, 1991.
[9] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.
[10] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *arXiv:1312.6034v2 [cs.CV]*, 2014.
[11] R. Caruana, H. Kangarloo, J. Dionisio, U. Sinha, and D. Johnson, "Case-based explanation of non-case-based learning methods," in *Proceedings of the AMIA Symposium*, 1999, pp. 212–215.
[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, 2013, pp. 3111–3119.
[13] B. Kim, C. Rudin, and J. A. Shah, "The bayesian case model: A generative approach for case-based reasoning and prototype classification," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 1952–1960.
[14] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, "Generating visual explanations," in *European Conference on Computer Vision*, 2016, pp. 3–19.
[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Computer Vision and Pattern Recognition (CVPR)*, 2015.
[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous systems," 2015, last accessed October 2017. [Online]. Available: http://tensorflow.org/
[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556
[18] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Mller, "Explaining nonlinear classification decisions with deep Taylor decomposition," *Pattern Recognition*, vol. 65, p. 211222, 2017.