Argflow: A Toolkit for Deep Argumentative Explanations for Neural Networks

Demonstration Track

Adam Dejl*
Imperial College London, UK
adam.dejl18@ic.ac.uk

Chloe He*
Imperial College London, UK
chloe.he18@ic.ac.uk

Pranav Mangal*
Imperial College London, UK
pranav.mangal18@ic.ac.uk

Hasan Mohsin*
Imperial College London, UK
hasan.mohsin18@ic.ac.uk

Bogdan Surdu*
Imperial College London, UK
george-bogdan.surdu18@ic.ac.uk

Eduard Voinea*
Imperial College London, UK
eduard-george.voinea18@ic.ac.uk

Emanuele Albini
Imperial College London, UK
emanuele.albini19@ic.ac.uk

Piyawat Lertvittayakumjorn
Imperial College London, UK
pl1515@ic.ac.uk

Antonio Rago
Imperial College London, UK
a.rago15@ic.ac.uk

Francesca Toni
Imperial College London, UK
ft@ic.ac.uk

ABSTRACT

In recent years, machine learning (ML) models have been successfully applied in a variety of real-world applications. However, they are often complex and incomprehensible to human users. This can decrease trust in their outputs and render their usage in critical settings ethically problematic. As a result, several methods for explaining such ML models have been proposed recently, in particular for black-box models such as deep neural networks (NNs). Nevertheless, these methods predominantly explain outputs in terms of inputs, disregarding the inner workings of the ML model computing those outputs. We present *Argflow*, a toolkit enabling the generation of a variety of *'deep' argumentative explanations (DAXs)* for outputs of NNs on classification tasks.

KEYWORDS

Computational Argumentation; Explainable AI; Neural Networks

ACM Reference Format:
Adam Dejl, Peter He, Pranav Mangal, Hasan Mohsin, Bogdan Surdu, Eduard Voinea, Emanuele Albini, Piyawat Lertvittayakumjorn, Antonio Rago, and Francesca Toni. 2021. Argflow: A Toolkit for Deep Argumentative Explanations for Neural Networks: Demonstration Track. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

Recently, machine learning (ML) has been successfully applied in a variety of real-world settings, including self-driving cars, automated translation, diagnostic engines, or job applicant screening. In many such deployments (e.g. in healthcare), understanding why certain outputs are generated can be critical. Explanations of ML systems may also be needed to assess the presence of algorithmic bias.

For some ML models, such as decision trees, generating explanations is relatively straightforward; one may say that they are *intrinsically interpretable*. However, for some ML models, and in particular those based on modern machine learning algorithms such as deep artificial neural networks (NNs), it is often difficult to understand why a certain output is generated, even for experts in ML. The development of methods and systems for extracting human-interpretable descriptions of black-box model behaviour such as NNs has thus recently received much attention in the field of explainable artificial intelligence (XAI), e.g. with *post-hoc* approaches for explanation. These include feature importance methods (such as LIME [8] and GradCAM [9]), prototype-based methods (such as activation maximisation [3]), model extraction (such as [2]) and counterfactual explanations (such as [10]). However, the majority of research has hitherto focused on explaining the output of machine learning models solely in terms of the input, without providing intuition regarding the models' inner workings.

Recently, a novel method of *deep argumentative explanations (DAXs)* has been proposed, drawing ideas from computational argumentation [1]. The advantage of DAXs over previous methods is that it constructs 'deep' explanations that reflect the internal influence structure of a model. In a convolutional neural network (CNN), this may correspond to how the detection of lower level features (such as linguistic or facial features) influence the detection of higher level features (such as text or face classification). Moreover, as the concepts of debating and argumentation are generally well-understood by human users, the explanations generated by computational argumentation can often be more intuitive than explanations generated using other methods. The overall DAX methodology is summarised in Figure 1. This involves constructing an *influence graph* (Step 1), converting it to a *generalised argumentation framework (GAF)* (Step 2) and then displaying the GAF to

*These authors contributed equally.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

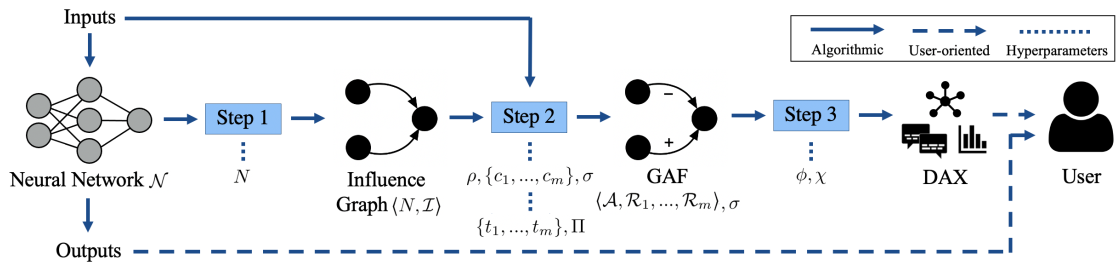

Figure 1: Adapted from [1]: DAX methodology (alongside the typical process of obtaining outputs from a neural model given inputs) comprising steps: 1. Based on chosen *nodes* N in $\mathcal{N}$, extract directed graph $\langle N, \mathcal{I} \rangle$ of *influences* between nodes; 2. Extract a *Generalised Argumentation Framework (GAF)* from the output of step 1, based on choices of *argument mapping* ρ, *dialectical relation characterisations* $\{c_1, ..., c_m\}$ and *dialectical strength* σ. These choices are driven by types $\{t_1, ..., t_m\}$ of dialectical relations to be extracted and *dialectical properties* Π that σ should satisfy (on the GAF); 3. Generate a *DAX* from the GAF and σ for user consumption in a certain *format* ϕ associating arguments with human-interpretable concepts through a *mapping* χ.

users in the relevant format with individual arguments visualised in a human-interpretable format (Step 3). We refer the reader to [1] for further details on the DAX methodology and its use of computational argumentation. Here, we provide an illustration of the DAX methodology and briefly overview the Argflow toolkit.

2 AN ILLUSTRATIVE EXAMPLE

Consider a CNN architecture [5] composed of an input layer taking word embeddings, a hidden convolutional layer, a max pooling layer, and finally a dense softmax layer. We can train this architecture on the *AG-News* dataset [4] to obtain a model for multi-class text classification. **Step 1.** We can choose $N = N_1 \cup N_2 \cup N_3$ with an input stratum N_1 with nodes corresponding to the input words, an intermediate stratum N_2 with nodes corresponding to the neurons of the max-pooling layer, and an output stratum $N_3 = \{n_o\}$ with n_o the neuron of the most probable class (for the given input). Influences can then be obtained from the connections between the nodes in the model. **Step 2.** We can choose to extract a GAF with two dialectical relations of types *attack* and *support* between arguments matching the strata in N. In particular, intermediate argument α_j represents (via hyperparameter ρ) filter $n_j \in N_2$ and input arguments α_{ij} represents (again via ρ) a word $n_i \in N_1$ that influences filter $n_j \in N_2$. An example GAF is given in Figure 2 (the outermost arguments represent the words in the input sentence and the innermost argument represent the predicted label *Business*). **Step 3.** We can choose to present DAXs in a variety of formats (determined by hyperparameter ϕ) from the GAF (see [1] for examples). Moreover, in order to render intermediate arguments (in this case, filters) in a manner comprehensible to humans, we can choose (through hyperparameter χ) to pair them with word clouds showing n-grams from the training set that activate the most the corresponding filter, as in [6].

3 ARGFLOW

We developed a generic toolkit for constructing DAXs for neural networks. The code is available at https://gitlab.com/argflow, and a video of experiments can be found at https://youtu.be/LPz4QbmLaxs

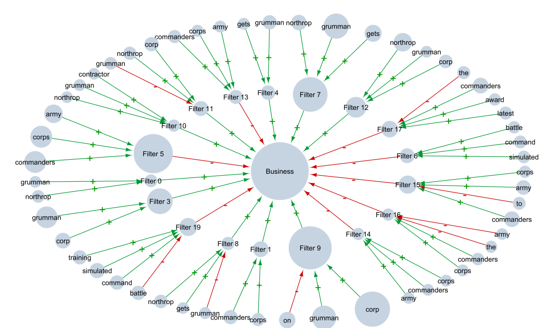

Figure 2: GAF (with attacks -; supports +) for a text classifier.

(where we present two demos generating explanations for VGG-16 [7] and a feed-forward NN). The design of Argflow is based on principles of modularity and extensibility, ensuring that it is flexible enough to be used for a variety of applications. The toolkit consists of a Python library for instantiating the hyperparameters and generating DAXs for a given model, and a web portal for delivering these DAXs to users with differing requirements.

Python Library. We collapse the first two steps into a single *GAF extraction step* handled by `GAFExtractor` class. This exposes a single `extract()` method which, given a model and its input, will return a GAF (represented by the `GAF` class) for the model run on its input. In order to visualise arguments in a human-interpretable modality, we provide the `Chi` abstract class. Argflow provides several out-of-the-box concrete implementations of the `Chi` class (GradCAM [9] and activation maximisation for convolutional filters). **Web Portal.** This provides users with the ability to visualise GAFs in different formats (ϕ). We use a typical web app architecture, with the frontend implemented as a React app using JavaScript, and a Python server for the application's backend. The portal provides a graphical interface to quickly import some classes of model and generate DAXs for them. However, this functionality can be extended with the `ExplanationGenerator` abstract class.

The visualisation system itself is customisable, but we provide two built-in visualisation types: graph-based and conversation-based.

REFERENCES

[1] Emanuele Albini, Piyawat Lertvittayakumjorn, Antonio Rago, and Francesca Toni. 2020. DAX: Deep Argumentative eXplanation for Neural Networks. *CoRR* abs/2012.05766 (2020). arXiv:2012.05766 https://arxiv.org/abs/2012.05766

[2] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. 2017. Interpreting Blackbox Models via Model Extraction. *CoRR* abs/1705.08504 (2017). http://arxiv.org/abs/1705.08504

[3] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2009. *Visualizing Higher-Layer Features of a Deep Network.* Technical Report 1341. University of Montreal. Also presented at the ICML 2009 Workshop on Learning Feature Hierarchies, Montréal, Canada.

[4] Antonio Gulli. 2005. AG-News Corpus. http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

[5] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *2014 Conf. on Empirical Methods in Natural Language Processing, EMNLP*. Association for Computational Linguistics, 1746–1751.

[6] Piyawat Lertvittayakumjorn, Lucia Specia, and Francesca Toni. 2020. FIND: Human-in-the-Loop Debugging Deep Text Classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*. 332–348. https://www.aclweb.org/anthology/2020.emnlp-main.24/

[7] Shuying Liu and Weihong Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. 730–734. https://doi.org/10.1109/ACPR.2015.7486599

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[9] Ramprassath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 618–626. https://doi.org/10.1109/ICCV.2017.74

[10] Sandra Wachter, Brent D. Mittelstadt, and Chris Russell. 2017. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *CoRR* abs/1711.00399 (2017). http://arxiv.org/abs/1711.00399