Computational Models of Argument
C. Reed et al. (Eds.)

doi:10.3233/FAIA240307

Graph Convolutional Networks and Graph Attention Networks for Approximating Arguments Acceptability

Paul CIBIER [a] and Jean-Guy MAILLY [b]
[a] *Independent researcher, Paris, France, paul.cibier@gmail.com*
[b] *IRIT, Université Toulouse Capitole, Toulouse, France jean-guy.mailly@irit.fr*

Abstract. Various approaches have been proposed for providing efficient computational approaches for abstract argumentation. Among them, neural networks have permitted to solve various decision problems, notably related to arguments (credulous or skeptical) acceptability. In this work, we push further this study in various ways. First, relying on the state-of-the-art approach `AFGCN`, we show how we can improve the performances of the Graph Convolutional Networks (GCNs) regarding both runtime and accuracy. Then, we show that it is possible to improve even more the efficiency of the approach by modifying the architecture of the network, using Graph Attention Networks (GATs) instead.

Keywords. Abstract Argumentation, Approximate Reasoning, Machine Learning, Arguments Acceptability

Computational methods for abstract argumentation [1] have received much attention in the last years, notably thanks to the organization of the International Competition on Computational Models of Argumentation (ICCMA).[1] Roughly speaking, there are two families of approaches: exact algorithms, which guarantee to find the correct result, but possibly require a large amount of time (because of the high computational complexity of many interesting problems [2]); and approximate algorithms, which typically outperform exact algorithms in terms of runtime, but may provide wrong results in some cases.

Several recent approaches have been proposed for defining approximate algorithms for abstract argumentation. Some of them rely on the relation between the grounded semantics and most other classical semantics [3], which allows to use this (polynomially computable) semantics as a good approximation for the status of most arguments under other (intractable) semantics. In the recent editions of ICCMA, these approaches are `HARPER++` [4] and `ARIPOTER` [5]. Another approach, named `FARGO-LIMITED` [6], is based on (bounded depth) DPLL-style algorithm for searching if an argument belongs to an admissible set, which allows to answer positively to various acceptability queries. Finally, using machine learning techniques for abstract argumentation has received some attention, in particular `AFGCN` [7] is based on Graph Convolutional Networks (GCNs). All these approaches participated to the last edition of ICCMA in 2023,[2] and solve credulous and skeptical acceptability problems, *i.e.* determining whether a given argument belongs to some or each extension for a given semantics.

[1] `http://argumentationcompetition.org`
[2] `http://argumentationcompetition.org/2023`

In this work, we explore two different ways to define approximate algorithms for these acceptability queries, both based on machine learning techniques. First, building upon AFGCN, we define new GCN-based methods for solving acceptability queries. We show that these methods, using different sets of features for training the model or different network architectures allow to outperform the original AFGCN solver. Then, we show how replacing GCNs by Graph Attention Networks (GATs) improves even more the performances of the solver.

1. Background

1.1. Abstract Argumentation

Definition 1 (Argumentation Framework [1])**.** An *Argumentation Framework* (AF) is a directed graph $\mathscr{F} = \langle \mathscr{A}, \mathscr{R} \rangle$ where $\mathscr{A}$ is a set of abstract entities called *arguments* and $\mathscr{R} \subseteq \mathscr{A} \times \mathscr{A}$ is the *attack relation*.

Although Dung does not put additional constraints regarding the set of arguments, in this work we assume that $\mathscr{A}$ is a non-empty finite set of arguments. When $(a,b) \in \mathscr{R}$, we say that *a attacks b*, and similarly if $\exists a \in S$ s.t. a attacks b, then the set of arguments S attacks b. Classical AF semantics rely on a notion of collective acceptability: the semantics allow to determine sets of *extensions*, which are sets of jointly acceptable arguments. Most extension-based semantics satisfy two basic properties:

Definition 2 (Conflict-freeness and defense)**.** Given an AF $\mathscr{F} = \langle \mathscr{A}, \mathscr{R} \rangle$ and the set of arguments $S \subseteq \mathscr{A}$, we say that S is *conflict-free* if $\forall a,b \in S$, $(a,b) \notin \mathscr{R}$. Then, given an argument $a \in \mathscr{A}$, we say that S *defends* a if $\forall b \in \mathscr{A}$ s.t. $(b,a) \in \mathscr{R}$, $\exists c \in S$ s.t. $(c,b) \in \mathscr{R}$.

A conflict-free set which defends all its elements is said *admissible*. Most semantics select extensions among the admissible sets of an AF.

Definition 3 (Dung's Semantics [1])**.** Given $\mathscr{F} = \langle \mathscr{A}, \mathscr{R} \rangle$ and $S \subseteq \mathscr{A}$ an admissible set,

- S is a *complete* extension iff it contains all the arguments that it defends,
- S is a *preferred* extension iff it is a $\subseteq$-maximal complete extension,
- S is a *grounded* extension iff it is a $\subseteq$-minimal complete extension,
- S is a *stable* extension iff it attacks every argument in $\mathscr{A} \setminus S$.

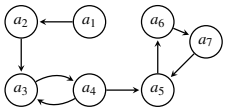
We write (resp.) $\mathbf{co}(\mathscr{F})$, $\mathbf{pr}(\mathscr{F})$, $\mathbf{gr}(\mathscr{F})$ and $\mathbf{st}(\mathscr{F})$ for the sets of complete, preferred, grounded and stable extensions of $\mathscr{F}$. It is known [1] that each AF has exactly one grounded extension, at least one preferred and complete extension, and $\mathbf{st}(\mathscr{F}) \subseteq \mathbf{pr}(\mathscr{F}) \subseteq \mathbf{co}(\mathscr{F})$. The status of an argument w.r.t. a given semantics is defined by:

Definition 4 (Argument Acceptability)**.** Given an AF $\mathscr{F} = \langle \mathscr{A}, \mathscr{R} \rangle$ and a semantics $\sigma \in \{\mathbf{co}, \mathbf{pr}, \mathbf{gr}, \mathbf{st}\}$, the argument $a \in \mathscr{A}$ is *credulously* (resp. *skeptically*) accepted if $a \in \mathbf{cred}_\sigma(\mathscr{F}) = \bigcup_{S \in \sigma(\mathscr{F})} S$ (resp. $a \in \mathbf{skep}_\sigma(\mathscr{F}) = \bigcap_{S \in \sigma(\mathscr{F})} S$).

Reasoning with the grounded semantics is tractable, but it is not the case in general for the other semantics. More precisely, **DC**-σ (resp. **DS**-σ) is the decision problem which consists in determining whether an argument a is credulously (resp. skeptically)

accepted w.r.t. the semantics σ. **DC**-σ is NP-complete for $\sigma \in \{\mathbf{co}, \mathbf{pr}, \mathbf{st}\}$, and **DS**-$\sigma$ is coNP-complete for $\sigma = \mathbf{st}$ and Π_2^P-complete for $\sigma = \mathbf{pr}$ (it is polynomial for $\sigma = \mathbf{co}$ because it coincides with **DS-gr**). See [2] for more details on this topic.

Example 1. Figure 1a depicts an AF $\mathscr{F}_1 = \langle \mathscr{A}_1, \mathscr{R}_1 \rangle$. Its extensions and the set of (credulously and skeptically) accepted arguments are given in Figure 1b.



(a) $\mathscr{F}_1$

Semantics σ	$\sigma(\mathscr{F}_1)$	$\mathbf{cred}_\sigma(\mathscr{F}_1)$	$\mathbf{skep}_\sigma(\mathscr{F}_1)$
co	$\{\{a_1,a_4,a_6\},\{a_1,a_3\},\{a_1\}\}$	$\{a_1,a_3,a_4,a_6\}$	$\{a_1\}$
pr	$\{\{a_1,a_4,a_6\},\{a_1,a_3\}\}$	$\{a_1,a_3,a_4,a_6\}$	$\{a_1\}$
gr	$\{\{a_1\}\}$	$\{a_1\}$	$\{a_1\}$
st	$\{\{a_1,a_4,a_6\}\}$	$\{a_1,a_4,a_6\}$	$\{a_1,a_4,a_6\}$

(b) Extensions and (credulously and skeptically) acceptable arguments

Figure 1. An AF $\mathscr{F}_1$ with its extensions and accepted arguments under $\sigma \in \{\mathbf{co}, \mathbf{pr}, \mathbf{gr}, \mathbf{st}\}$.

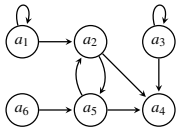
While extension-based semantics rely on a notion of collective acceptability of arguments (and then, individual acceptability can be derived thanks to credulous or skeptical reasoning), other families of semantics directly capture a notion of individual acceptability. In particular, a *gradual semantics* assigns to each argument in an AF an *acceptability degree*, usually a real number in the interval $[0,1]$, where 0 represents complete rejection of the argument and 1 complete acceptance. These semantics are based on different intuitions, for instance taking into account the number or the quality of an argument's attackers. To represent the attackers of an argument a, we define $a^- = \{b \in \mathscr{A} \mid (a,b) \in \mathscr{R}\}$. Here are some classical gradual semantics that we use in this work.

Definition 5 (Gradual Semantics)**.** Given an AF $\mathscr{F} = \langle \mathscr{A}, \mathscr{R} \rangle$ and $a \in \mathscr{A}$, the gradual semantics *h-categorizer* (**h − cat** [8]), *no self-attacker* (**nsa** [9]), *Max-based* (**Mbs** [10]) and *Card-based* (**Cbs** [10]) map each argument $a \in \mathscr{A}$ to a value in $[0,1]$ as follows:

$$\mathbf{h-cat}(\mathscr{F},a) = \frac{1}{1+\sum_{b\in a^-} \mathbf{h-cat}(\mathscr{F},b)} \qquad \mathbf{nsa}(\mathscr{F},a) = \begin{cases} 0 \text{ if } (a,a) \in \mathscr{R} \\ \frac{1}{1+\sum_{b\in a^-} \mathbf{nsa}(\mathscr{F},b)} \text{ otherwise} \end{cases}$$

$$\mathbf{Mbs}(\mathscr{F},a) = \frac{1}{1+\max_{b\in a^-} \mathbf{Mbs}(b,\mathscr{F})} \qquad \mathbf{Cbs}(\mathscr{F},a) = \frac{1}{1+|a^-|+\frac{\sum_{b\in a^-} \mathbf{Cbs}(\mathscr{F},b)}{|a^-|}}$$

Example 2. Let $\mathscr{F}_2 = \langle \mathscr{A}_2, \mathscr{R}_2 \rangle$ be the AF from Figure 2a. For each $x \in \mathscr{A}_2$, we give the acceptability degree for all the gradual semantics considered in the paper in Table 2b.



(a) $\mathscr{F}_2$

Argument x	$\mathbf{h-cat}(\mathscr{F}_2,x)$	$\mathbf{nsa}(\mathscr{F}_2,x)$	$\mathbf{Mbs}(\mathscr{F}_2,x)$	$\mathbf{Cbs}(\mathscr{F}_2,x)$
a_1	0.618	0.0	0.618	0.414
a_2	0.495	0.732	0.618	0.299
a_3	0.618	0.414	0.618	0.414
a_4	0.398	0.477	0.618	0.231
a_5	0.401	0.399	0.5	0.274
a_6	1.0	1.0	1.0	1.0

(b) Acceptability degrees (rounded to 10^{-3})

Figure 2. An AF $\mathscr{F}_2$ and the acceptability degrees of arguments for $\sigma \in \{\mathbf{h-cat}, \mathbf{nsa}, \mathbf{Mbs}, \mathbf{Cbs}\}$.

1.2. Neural Networks for Argumentation

Now we present basic notions of deep learning as used in the rest of this paper. Deep learning is an efficient way to approximate the solutions to various kinds of problems [11]. Given the high complexity of most interesting problems in abstract argumentation [2], it is not surprising that several approximate reasoning approaches have been proposed for argumentation, including some based on deep learning.

An artificial neuron (or simply, a neuron from now) can be seen as a computational unit taking as input a vector of values $x = (x_1, \ldots, x_n)$, a vector of weights $w = (w_1, \ldots, w_n)$ and an additional information called the bias b. The neuron is mainly an *activation function* f, and the output of the neuron is obtained by applying f to the weighted sum of the input values and the bias, *i.e.* $f(b + \sum_{i=1}^{n} w_i \times v_i)$. Neurons are connected into networks, usually made of various *layers*. The *input layer* is made of neurons that only pass their input values to the next layer without processing (*i.e.* there is no activation function), then there are (possibly many) hidden layers and finally one output layer, s.t. the output of these last neurons correspond to the output of the neural network. For training a neural network in a context of supervised learning, the output of the network is compared to the labels of the training data, and a back-propagation algorithm is used to find the most accurate representation of the data by adjusting the weights of the neurons. This back-propagation aims at minimizing the value of the loss function, which represents the distance between the value predicted by the network output and the real value of the labeled data. This allows to provide a better approximation of the mapping (input data $\rightarrow$ labels) by the neural network. Neural networks are typically represented as (weighted) directed graphs where the nodes are the neurons and the weighted edges are the connections between neurons.

1.2.1. Graph Convolutional Network

While "basic" neural networks as described before take a vector of real numbers as input, [12] introduces neural networks able to directly use graphs as their inputs, named Graph Convolutional Networks (GCNs). A GCN works by taking as input an adjacency matrix representation of the graph and a node embedding, *i.e.* a set of features for each node of the graph. A layer of a GCN works by applying a convolution on each dimension of the vector representing the node embedding. The convolution takes into account the node embedding of the neighbours of the node, and the node itself. Formally, from a layer l of the GCN and the adjacency matrix of the graph A, the layer $l+1$ can be obtained by computing $H^{l+1} = f(H^l, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^l W^l)$ where $\hat{A}$ is the representation of the graph where self-loops have been added on each argument, W^l is the matrix of weights for the layer l and σ is a (non-linear) activation function. The self-loops allow a node of the argumentation framework to propagate information to itself (otherwise, the node would only receive information about its neighbours). The first layer H^1 can be obtained by computing $f(H^0, A)$ where H^0 is a matrix representation of the node features.

The GCN model was used in the context of abstract argumentation for approximating the credulous or skeptical acceptability of arguments under various semantics. [13] focused on credulous acceptability under the preferred semantics (**DC-pr**) and proposed two different node embeddings. In the first one, each node is associated to a single feature, which is the same constant for every node (*i.e.* there is no additional information provided for the nodes), while the second one provides two features per node, namely

the numbers of incoming and outgoing attacks. Then, another approach based on GCNs has been proposed, with the solver AFGCN performing extremely well at ICCMA 2021.[3] The second version of AFGCN, which participated to ICCMA 2023, can solve both **DC** and **DS** problems for the semantics **co**, **pr** and **st**, as well as other semantics not considered in this paper (semi-stable, stage and ideal [14]). We give more details on AFGCN in Section 2, where we describe the various modifications that we have made in order to improve its efficiency regarding both runtime and precision.

1.2.2. Graph Attention Network

In classical graph neural networks like GCNs, the feature update of a node when passing from layer l to layer $l+1$ is typically an average of the features of all its neighbors and itself. It means that there is no difference in the treatment of all neighbors. With Graph Attention Networks (GATs) [15], there is an assumption that some node are more important than some other ones. To take into account this assumption, when updating the node embedding of the node in a graph attentional layer, an attention score is computed between the source node and every neighbour of this node (*e.g.* using a weighted matrix applied to the node embedding of the nodes, the function `Leaky ReLU`, normalized by using the function `softmax`), which allows to indicate if a node j is important or not for a node i. These attention coefficients are used to weigh the influence of a neighbors' features when computing the new embedding of the node during the convolution. The authors of [15] also incorporate the mechanism of multi-head attention, which allows the network to perform this process several times independently and in parallel. When finishing the computation of the updated features with each attention head, the results of all the heads are concatenated (for inner layers) or averaged (for output layers).

2. New Neural Networks for Arguments Acceptability

We propose several new approaches to (approximatively) evaluate arguments acceptability in abstract argumentation. Section 2.1 shows our modified version of the AFGCN approach, and Section 2.2 shows our proposal for using GATs instead of GCNs. Our software is available here: https://github.com/Paulo-21/AF-GCN-GAT_wGS .

2.1. GCNs for Arguments Acceptability

Our first approach is based on the work by Lars Malmqvist using Graph Convolutional Networks [7] for approximating arguments acceptability, but (among other technical differences) our method uses a different kind of node embedding. Based on [7] and the solver source code[4], let us first describe with more details the AFGCN solver, more precisely its second version that was submitted to ICCMA 2023. AFGCNv2 starts by running a grounded extension solver (based on NumPy [16], which provides a quadratic space representation of the AF), and if the query argument is not a member of the grounded extension, then a GCN is used to predict the acceptability of this argument. The GCN model is built as follow. Its inputs are the adjacency matrix of the argumentation graph

[3] http://argumentationcompetition.org/2021/
[4] https://github.com/lmlearning

and a graph embedding made of a vector of dimension 128 for each argument, containing features representing the eigenvector centrality, the graph centrality, the in-degree and out-degree, PageRank and the graph coloring. The other features are randomized input features. All of these assign numbers to the nodes, which are normalized in order to belong to a comparable scale and constitute the node embedding of the graph. The core component of the GCN is built with 4 consecutive blocks, each made of a GCN layer and a Dropout layer [17], which aim at reducing the risk of overfitting by dropping some of the neurons during the forward pass of the learning process. At each block, there are residual connections that feed it with the original features (in addition to the new embedding produced by the previous layer). After these blocks, a layer reduces the dimension of the features vector to 1. Finally, a Sigmoid layer computes the probability for the arguments to be accepted.

2.1.1. Speed and Memory Management Improvements

There are several parts of the process where AFGCNv2 could not succeed its computation because of output errors or timeout, notably when managing large instances.

Python provides various interesting tools for applying machine learning techniques, but it may lack of efficiency in some cases. In particular, for AFGCNv2, the time taken for computing the graph centrality when building the node embedding may be important, as well as the time required to parse the text files describing large instances.

To improve the speed of the solver, we propose a Rust Made Python package that can read and parse the AF much faster than the Python implementation. This package receives a path to the file describing the AF, and then parses it. The second part of the process is the computation of its grounded extension. This Rust tool verifies if the query argument belongs to the grounded extension or is attacked by it. In this case, the system immediately stops and provides the correct answer. If the argument is neither in the grounded extension nor attacked by it (*i.e.* it is labeled UNDEC regarding the grounded labelling of the AF [18]), then we need to use the neural network to estimate the acceptability of the arguments. So, the Rust package computes the node embedding of the argumentation graph in order to feed the GCN or kind of GNN that need a node embedding.

This approach can speed up a lot the process for several reasons. As said before, the Rust package is faster for parsing large AF files compared to the original Python implementation. The main improvement regarding runtime is computation of the graph centrality, which was the main cause for long runtimes (and timeout) with AFGCNv2. Its computation is much faster with the Rust package than with the Python-based NetworkX library [19] used by AFGCNv2. The other potential runtime improvement concerns arguments attacked by the grounded extension. These arguments cannot belong to any extension under any semantics considered in this paper (nor other classical semantics like the semi-stable and ideal semantics [14]), but AFGCNv2 still needs computing the node embedding and running the GCN for them, while our package instantly answers that these arguments are not acceptable. This approach, inspired by [4,5] can considerably speed up the computation for arguments attacked by the grounded extension.

Another issue with the computation of the grounded extension is the use of the quadratic data structure in AFGCNv2. This leads to some *out-of-memory* errors with large instances (for instance, an AdmBuster graph[5] with $n = 50000$, requires 18.6 GiB of mem-

[5] See https://argumentationcompetition.org/2017/AdmBuster.pdf for details on AdmBuster.

ory). On the contrary, our Rust package relies on a linear algorithm and data structure, similar to the approach from [20], which avoids errors like those faced by AFGCNv2.

2.1.2. Model Precision Improvement

The node embedding of a graph maps each node (in our case, each argument) of the graph with a vector of N dimensions, each dimension representing information about some features of the node. As explained previously, in AFGCN, $N = 128$ with a large part of the features being randomly generated. We propose several variants of the node embedding of AFGCN, where the acceptability degrees of the argument for various gradual semantics (namely, $\mathbf{h-cat}$, **nsa**, **Cbs** and **Mbs**) are added to the features of the argument. Another feature that we take into account is the acceptability status w.r.t. the grounded semantics (represented as 1 for arguments in the grounded extension, 0 for arguments attacked by it, and 0.5 for the remaining arguments). Now we describe five versions of AFGCN:

- AFGCNv2, *i.e.* the solver that participated to ICCMA 2023, (with the features corresponding to the eigenvector centrality, the graph centrality, the in-degree and out-degree, PageRank, the graph coloring, and the randomised values),
- AFGCN-P128, our implementation (with the technical improvements described in Section 2.1.1) with the 6 features used in AFGCNv2 and the 5 additional features corresponding to the gradual semantics and the grounded semantics, and 117 randomly generated features,
- AFGCN-P11, our implementation with only the 11 "meaningful" features (*i.e.* without the random features).
- AFGCN-P128^{-do}, similar to AFGCN-P128 but without dropout layers,
- AFGCN-P11^{-do}, similar to AFGCN-P11 but without dropout layers.

Comparing AFGCNv2 with AFGCN-P128 aims at evaluating the interest of the new features (*i.e.* the gradual semantics and the grounded semantics), while the comparison between AFGCN-P128 and AFGCN-P11 assesses the impact of the randomized features. These randomized features are used for preventing the dropout layers from having a negative impact on the learning process by dropping too many important features. For this reason, we also perform a comparison of these approaches without the dropout layers.

2.2. GATs for Arguments Acceptability

We have also implemented a Graph Attention Network (called AFGAT in the rest of the paper), and used the second version proposed by the authors [15], which generally outperforms the first version in all benchmarks according to the authors of the model. Intuitively, not all neighbours of an arguments have the same impact on its acceptability (*e.g.* an attacker of a which is attacked by the grounded extension has no impact on a, but an attacker which is UNDEC in the grounded labelling may belong to some extension and thus prevent a from being skeptically accepted), hence the interest of the attention mechanism of GATs for evaluating arguments acceptability. As far as we know, it is the first implementation of a GAT for argument acceptability according to extension-based semantics. In our experiment we choose to implement the GATv2 with 3 graph attentionals layers parameterized with 5 head for the first layer, and 3 heads for the second and the third ones. We only consider the same 11 input features as in AFGCN-P11. For the following layers, the numbers of their input features is the number of output features

from the previous layer multiplied by the number of attention heads from the previous layer (so there are $5 \times 5 = 25$ features for the second layer, and $5 \times 3 = 15$ for the third layer), while the output layer only has one feature representing the acceptability of the arguments we have an Sigmoid Layer to transform the output feature of the last layer into a probability of acceptability. Following the approach described by [15], the concatenation of the results provided by each attention head is used between the layers 1 and 2 on the one hand, and 2 and 3 on the other hand. The average operation is used after the last layer (instead of concatenation) in order to obtain a single value for each argument.

3. Experimental Evaluation

In the following, we compare the efficiency of our approaches with those of `HARPER++` and `FARGO-LIMITED` which were the best performers in the approximate track of ICCMA 2023, as well as `AFGCN` which is the state-of-the-art approximate solver based on neural networks (and the foundation of our work).

3.1. Protocol

All our GNNs (*i.e.* `AFGCN-P11`, `AFGCN-P128`, `AFGCN-P11`$^{-do}$, `AFGCN-P128`$^{-do}$ and `AFGAT`) were trained with the same dataset used to train `AFGCNv2` by Lars Malmqvist, *i.e.* an aggregate of instances from ICCMA 2017, 900 AFs from the sets A, B, C[6]. Similarly to `AFGCNv2`, we used the Deep Graph Library (DGL) and Pytorch to train our models, with the option "shuffle" enabled to allow DGL to reshuffle the set of graphs given to the GNN for every batch during training. The batch size was set to 64 for all variants of `AFGCN` and 4 for `AFGAT` (because of memory limitations). For the optimization process, we used Adam [21] (like in `AFGCNv2`). The learning rate was set to 0.01, like `AFGCNv2`. The training ended after 400 epochs. The hardware used for the training was a CPU i3-9100F, 8GB of RAM and a GPU Nvidia GTX 1070 (8GB VRAM). For the test we used the same configuration except that solvers only had access to the CPU for this phase.

3.2. Results

Now, we describe our experimental results regarding four classical decision problems, namely **DC-co**, **DC-st**, **DS-pr** and **DS-st**. We conducted two different evaluations, using the instances from the ICCMA 2023 competition (329 instances).[7]

3.2.1. Comparison of the Neural Networks

In this first experiment, in order to define our test set we tried to compute with the SAT-based solver Crustabri (see `https://github.com/crillab/crustabri`) the acceptability status for each argument in each AF in the dataset. Notice that in this case, we only compare the relative efficiency of the neural networks, but we do not use the pre-computation of the grounded semantics which is mentioned in Section 2.1.1. Some instances required too much time to do so, and were thus excluded from the test set. Since

[6] `https://github.com/lmlearning/AFGraphLib`
[7] `https://zenodo.org/records/8348039/files/iccma2023_benchmarks.zip`

all problems are not equivalently hard for a given AF, the exact test set is slightly different for each of them (**DC-co**: 252 instances, **DC-st**: 268 instances, **DS-pr**: 217 instances and **DS-st**: 259 instances). Then, for each of these problems and AFs, we have executed the six different GNN models. When running a GCN or a GAT, the output layer of the neural network does not only give an estimation of the acceptability for one specific argument, but for all the arguments in the AF, which means we can compare the precision of these approaches for all the arguments. A possible issue here, if we chose to directly present global results (for instance, providing the accuracy as n/m with n the number of correctly predicted arguments, and m the total number of arguments over the full test set), would be related to the variability in the sizes of the AFs. For instance, if a neural network is particularly efficient for a given type of AFs which are generally large, it could "hide" the fact that the network does not perform very well on another type of AFs which are typically small. So we first compute the accuracy on a given instance passed to the GNN, where all its arguments are taken into account ($\theta_{\mathscr{F}} = n/m$ with n the number of correctly predicted arguments in $\mathscr{F}$, and m the number of arguments in $\mathscr{F}$). From this accuracy of the predicted acceptability on one given instance, we compute the average accuracy across all AFs to get the global accuracy ($\sum_{\mathscr{F}} \theta_{\mathscr{F}}/N$ with N the number of AFs).

GNN	**DC-co**	**DC-st**	**DS-pr**	**DS-st**
AFGCNv2	53 (48;76)	75 (57;76)	91 (20;**99**)	64 (17;98)
AFGCN-P128	70.9 (38;95)	83 (65;89)	96 (52;**99**)	71 (47;99)
AFGCN-P11	67 (28;**96**)	73 (42;90)	92 (26;97)	69 (31;99)
AFGCN-P128^{-do}	76 (52;89)	82 (61;89)	97 (63;**99**)	72 (51;99)
AFGCN-P11^{-do}	74 (46;93)	83.0 (62;88)	96 (56;**99**)	71 (47;98)
AFGAT	**88** (**78**;90)	**87** (**77**;**92**)	**98** (**80**;90)	**73** (**59**;**99**)

Table 1. Accuracy of the GNNs. In each cell, the first value corresponds to the global accuracy, and the number between parenthesis correspond respectively to the accuracy restricted to the positive and negative instances. Bold-faced values represent the highest accuracy for a problem (and a type of instances).

Our results are given in Table 1. The main insight that we obtain from this experiment is the overall performance of AFGAT, which obtains the best highest accuracy in most of cases. We also observe that (except for **DC-co**), our AFGCN-P128 performs better than AFGCNv2, which confirms the interest of our new features in the learning process. On the contrary, the version without randomized features (AFGCN-P11) is the least performing approach. The most plausible explanation is that dropout layers induce a major loss of information on a small number of features. This also explains why both approaches without dropout layers perform better than the other solvers based on GCNs.

3.2.2. Comparison with the ICCMA 2023 Participants

For the second experiment, we follow more closely the process of ICCMA 2023. In this case, for each AF, we use only one query argument for each AF (the same that was used in the competition). For this reason, there is no need to exclude AFs from the test set, since the ground truth is provided by ICCMA 2023 organizers. This means that we can easily compare the results of an approach based on neural networks (here, AFGAT) with the results of the best performers during the competition, namely FARGO-LIMITED and HARPER++. In this experiment, a timeout of 39 seconds was enforced (instead of

60 seconds at ICCMA 2023, because the experiments we conducted on a CPU 54% higher clock rate than ICCMA 2023), and the pre-processing based on the grounded semantics is incorporated in all our GNNs based solver. The number of correctly solved instances are given in Table 2. We observe that FARGO-LIMITED generally outperforms the other approaches, except for the skeptical acceptability under the stable semantics where our AFGAT obtains the best results. However, the results are generally tight, and let us envision better results for future versions of AFGAT with other parameters or another training dataset used in the learning process.

GNN	DC-co	DC-st	DS-pr	DS-st
AFGAT	265	268	297	**205**
HARPER++	220	187	300	196
FARGO-LIMITED	**300**	**307**	**303**	199

Table 2. Number of correctly solved instances by the solvers on the ICCMA 2023 test set.

4. Related Work

Besides AFGCN, other approaches have been proposed for approximate reasoning in abstract argumentation, including some based on machine learning. As mentioned earlier, [13] proposed a first implementation of a GCN for solving the credulous acceptability problem under the preferred semantics. This preliminary work showed the potential interest of using GCNs for argumentation, although the approach did not exhibit high enough performances for practical application (with accuracy scores remaining low for some types of instances). Then, [22,23] proposed a so-called Argumentation Graph Neural Network (AGNN) which approximates the acceptability of arguments under several semantics with a high accuracy. The main difference between this approach and most other approaches (which participate to ICCMA competitions) is the size of the instances (less than 200 arguments). Studying whether the AGNN approach scales-up well with ICCMA instances (w.r.t. runtime and accuracy), and how it compares with our approaches, is an interesting task for future work. Deep learning has also been used for other purposes, like extension enforcement [24,25] or approximating gradual semantics of Bipolar AFs [26]. Even if these contributions are out of the scope of the present paper, a deeper analysis of these works could provide interesting insights for improving our algorithms. Finally, we have already briefly mentioned another approach for approximate reasoning that participated to ICCMA 2023, namely ARIPOTER [5]. This solver combines the initial intuition of HARPER++ (regarding the grounded extension) and the use of gradual semantics for evaluation arguments acceptability. In ICCMA 2023, it performed generally better than AFGCN but not as good as HARPER++ and FARGO-LIMITED. An experimental comparison of our approaches with ARIPOTER is also an interesting future work.

5. Conclusion

We have shown how small changes of the GCN architecture from AFGCN can improve its performance on ICCMA 2023 benchmarks. Also, we have proposed a new approach

based on GATs, which performs even better than all the GCN variants that we have tested, and is competitive with the best performers from ICCMA 2023. This opens interesting research questions regarding the used of neural networks in abstract argumentation. A first natural extension of this work consists of empirically evaluating other variants of our GNN models (*e.g.* by testing other parameters for the behaviour of the dropout layers in the GCNs or the graph attentional layers in the GATs). We plan to continue this line of work, *e.g.* determining how our approaches generalize to other kinds of instances, or how other approaches based on neural networks compare to ours w.r.t. accuracy or computation time. We also wish to extend our approach by taking into account other semantics (*e.g.* the ideal, semi-stable or stage semantics) or by considering other kinds of abstract argumentation frameworks, like Incomplete AFs [27] for which reasoning may be harder than standard AFs. Finally, it would be interesting to provide approximate reasoning methods for other problems (*e.g.* computing some extension of an AF).

Acknowledgement

The second author received funding from the French National Research Agency (ANR grants ANR-22-CE23-0005 and ANR-22-CPJ1-0061-01). The authors thank Jérôme Delobelle for providing Example 2.

References

[1] Dung PM. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. Artif Intell. 1995;77(2):321-58. Available from: https://doi.org/10.1016/0004-3702(94)00041-X.

[2] Dvorák W, Dunne PE. Computational Problems in Formal Argumentation and their Complexity. In: Handbook of Formal Argumentation. College Publications; 2018. p. 631-88.

[3] Cerutti F, Thimm M, Vallati M. An experimental analysis on the similarity of argumentation semantics. Argument Comput. 2020;11(3):269-304. Available from: https://doi.org/10.3233/AAC-200907.

[4] Thimm M. Harper++ v1.1.1. In: Fifth International Competition on Computational Models of Argumentation (ICCMA'23); 2023. Available from: http://www.mthimm.de/pub/2023/Thimm_2023c.pdf.

[5] Delobelle J, Mailly JG, Rossit J. Revisiting Approximate Reasoning Based on Grounded Semantics. In: Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 17th European Conference, ECSQARU 2023. vol. 14294 of Lecture Notes in Computer Science. Springer; 2023. p. 71-83. Available from: https://doi.org/10.1007/978-3-031-45608-4_6.

[6] Thimm M. Fargo-limited v1.1.1. In: Fifth International Competition on Computational Models of Argumentation (ICCMA'23); 2023. Available from: http://www.mthimm.de/pub/2023/Thimm_2023b.pdf.

[7] Malmqvist L, Yuan T, Nightingale P, Manandhar S. Determining the Acceptability of Abstract Arguments with Graph Convolutional Networks. In: Third International Workshop on Systems and Algorithms for Formal Argumentation co-located with the 8th International Conference on Computational Models of Argument (COMMA 2020). vol. 2672 of CEUR Workshop Proceedings. CEUR-WS.org; 2020. p. 47-56. Available from: https://ceur-ws.org/Vol-2672/paper_5.pdf.

[8] Besnard P, Hunter A. A logic-based theory of deductive arguments. Artif Intell. 2001;128(1-2):203-35.

[9] Beuselinck V, Delobelle J, Vesic S. A Principle-based Account of Self-attacking Arguments in Gradual Semantics. J Log Comput. 2023;33(2):230-56.

[10] Amgoud L, Ben-Naim J, Doder D, Vesic S. Acceptability Semantics for Weighted Argumentation Frameworks. In: Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017; 2017. p. 56-62. Available from: https://doi.org/10.24963/ijcai.2017/9.

[11] Dong S, Wang P, Abbas K. A survey on deep learning and its applications. Computer Science Review. 2021;40:100379. Available from: https://www.sciencedirect.com/science/article/pii/S1574013721000198.

[12] Kipf TN, Welling M. Semi-Supervised Classification with Graph Convolutional Networks. In: 5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings. OpenReview.net; 2017. Available from: https://openreview.net/forum?id=SJU4ayYgl.

[13] Kuhlmann I, Thimm M. Using Graph Convolutional Networks for Approximate Reasoning with Abstract Argumentation Frameworks: A Feasibility Study. In: Scalable Uncertainty Management - 13th International Conference, SUM 2019. vol. 11940 of Lecture Notes in Computer Science. Springer; 2019. p. 24-37. Available from: https://doi.org/10.1007/978-3-030-35514-2_3.

[14] Baroni P, Caminada M, Giacomin M. Abstract Argumentation Frameworks and Their Semantics. In: Handbook of Formal Argumentation. College Publications; 2018. p. 159-236.

[15] Velickovic P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph Attention Networks. In: 6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings. OpenReview.net; 2018. Available from: https://openreview.net/forum?id=rJXMpikCZ.

[16] Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020 Sep;585(7825):357-62. Available from: https://doi.org/10.1038/s41586-020-2649-2.

[17] Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15(1):1929-58. Available from: https://dl.acm.org/doi/10.5555/2627435.2670313.

[18] Caminada M. On the Issue of Reinstatement in Argumentation. In: Logics in Artificial Intelligence, 10th European Conference, JELIA 2006. vol. 4160 of Lecture Notes in Computer Science. Springer; 2006. p. 111-23. Available from: https://doi.org/10.1007/11853886_11.

[19] Hagberg AA, Schult DA, Swart PJ. Exploring Network Structure, Dynamics, and Function using NetworkX. In: 7th Python in Science Conference; 2008. p. 11 15.

[20] Nofal S, Atkinson K, Dunne PE. Computing Grounded Extensions Of Abstract Argumentation Frameworks. Comput J. 2021;64(1):54-63. Available from: https://doi.org/10.1093/comjnl/bxz138.

[21] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings; 2015. Available from: http://arxiv.org/abs/1412.6980.

[22] Craandijk D, Bex F. AGNN: A Deep Learning Architecture for Abstract Argumentation Semantics. In: Computational Models of Argument - Proceedings of COMMA 2020. IOS Press; 2020. p. 457-8. Available from: https://doi.org/10.3233/FAIA200532.

[23] Craandijk D, Bex F. Deep Learning for Abstract Argumentation Semantics. In: Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020. ijcai.org; 2020. p. 1667-73. Available from: https://doi.org/10.24963/ijcai.2020/231.

[24] Craandijk D, Bex F. EGNN: A Deep Reinforcement Learning Architecture for Enforcement Heuristics. In: Computational Models of Argument - Proceedings of COMMA 2022. vol. 353 of Frontiers in Artificial Intelligence and Applications. IOS Press; 2022. p. 353-4. Available from: https://doi.org/10.3233/FAIA220169.

[25] Craandijk D, Bex F. Enforcement Heuristics for Argumentation with Deep Reinforcement Learning. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022. AAAI Press; 2022. p. 5573-81. Available from: https://doi.org/10.1609/aaai.v36i5.20497.

[26] Al Anaissy C, Suntwal S, Surdeanu M, Vesic S. On Learning Bipolar Gradual Argumentation Semantics with Neural Networks. In: 16th International Conference on Agents and Artificial Intelligence, ICAART 2024, Volume 2; 2024. p. 493-9. Available from: https://doi.org/10.5220/0012448300003636.

[27] Mailly JG. Yes, no, maybe, I don't know: Complexity and application of abstract argumentation with incomplete knowledge. Argument Comput. 2022;13(3):291-324. Available from: https://doi.org/10.3233/AAC-210010.