Deep Learning for Abstract Argumentation Semantics

Dennis Craandijk[1,2] and **Floris Bex**[2,3]
[1] National Police Lab AI, Netherlands Police
[2] Information and Computing Sciences, Utrecht University
[3] Institute for Law, Technology and Society, Tilburg University
{d.f.w.craandijk, f.j.bex}@uu.nl

Abstract

In this paper, we present a learning-based approach to determining acceptance of arguments under several abstract argumentation semantics. More specifically, we propose an argumentation graph neural network (AGNN) that learns a message-passing algorithm to predict the likelihood of an argument being accepted. The experimental results demonstrate that the AGNN can almost perfectly predict the acceptability under different semantics and scales well for larger argumentation frameworks. Furthermore, analysing the behaviour of the message-passing algorithm shows that the AGNN learns to adhere to basic principles of argument semantics as identified in the literature, and can thus be trained to predict extensions under the different semantics – we show how the latter can be done for multi-extension semantics by using AGNNs to guide a basic search. We publish our code at https://github.com/DennisCraandijk/DL-Abstract-Argumentation.

1 Introduction

Over the past few years an increasing amount of research effort has been directed towards designing deep learning models that learn on problems from symbolic domains [d'Avila Garcez *et al.*, 2015]. Recent progress has sparked interest in graph neural networks (GNNs), a class of neural networks capable of performing computations over graphs. Due to their strong relational inductive bias [Battaglia *et al.*, 2018], GNNs can be trained to solve constraint satisfaction problems which require performing relational inferences such as boolean satisfiability [Selsam *et al.*, 2019] and solving Sudoku puzzles [Palm *et al.*, 2018].

One domain in symbolic AI that is relatively unexplored with respect to GNNs is *computational argumentation*, an approach to defeasible reasoning that focuses on interactions between arguments and counterarguments. With applications in multi-agent systems, decision-making tools, medical and legal-reasoning, argumentation has become a major subfield of AI [Atkinson *et al.*, 2017]. Much of the theory in computational argumentation is built on Dung's [1995] pioneering work on abstract argumentation frameworks, which introduced several acceptability semantics that define which sets of arguments (*extensions*) can be reasonably accepted given an argumentation framework (AF) of arguments and attacks between these arguments, often represented as a graph. Thus, it can be determined if an argument can be accepted given an AF by looking at whether it is contained in some extensions (credulous acceptance) or all extensions (sceptical acceptance) under a given semantics. Due to the computational complexity of determining which arguments can be accepted, the design of efficient methods for computing extensions and acceptability constitutes an active research direction within the argumentation community. Most current approaches solve acceptance problems by translating the problem to a symbolic formalism for which a dedicated solver exists, such as constraint-satisfaction problems, propositional logic or answer-set programming [Gaggl *et al.*, 2020; Charwat *et al.*, 2015].

In this paper, we propose an argumentation graph neural network (AGNN) that learns to predict credulous and sceptical acceptance of arguments under 4 well-known argumentation semantics. To the best of our knowledge, only Kulhman and Thimm [2019] have conducted a preliminary study using GNNs, implementing a conventional single forward pass classifier to approximate credulous acceptance under the preferred semantics with an average class accuracy of around 0.61. We propose a recurrent GNN that can almost perfectly predict (MCC between 0.997 and 1) both credulous and sceptical acceptance under several semantics by learning to perform a sequence of relational inferences based on the attack relations between the arguments in an AF. Furthermore, we also provide a way to predict (multiple) extensions given an AF by using AGNN to guide a search procedure.

Our learning-based approach to determining argument acceptance shows that sub-symbolic deep learning techniques can accurately solve a problem that could previously only be solved by sophisticated symbolic solvers. By inspecting the behaviour of the message-passing algorithm of AGNN, we see that it has learnt some basic principles of argumentation semantics [Baroni *et al.*, 2011], and in the case of acceptance under the grounded semantics exhibits behaviour similar to a well-established symbolic labelling algorithm for the grounded extension [Modgil and Caminada, 2009]. AGNN is a single architecture that is able to approximate argumentation problems of differing complexity and of sizes sub-

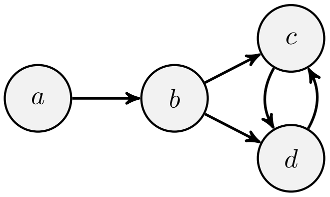

Figure 1: Graph representations of the AF F_e.

stantially larger than what it saw during training in constant time, simply by running for more iterations. While symbolic solvers always provide the correct answer, different problems under different semantics each need their own tailor-made algorithm, and the time complexity depends on the complexity of the problem.

2 Preliminaries

We recall Dung's abstract argumentation frameworks [1995].

Definition 1. *An abstract* argumentation framework *(AF) is a pair $F = (A, R)$ where A is a (finite) set of arguments and $R \subseteq A \times A$ is the attack relation. The pair $(a, b) \in R$ means that a* attacks *b. A set $S \subseteq A$ attacks b if there is an $a \in S$, such that $(a, b) \in R$. An argument $a \in A$ is* defended *by $S \subseteq A$ iff, for each $b \in A$ such that $(b, a) \in R$, S attacks b.*

Example 1. *Figure 1 (a) illustrates the AF $F_e = (\{a, b, c, d\}, \{(a, b), (b, c), (b, d), (c, d), (d, c)\})$, which serves as a running example.*

Dung-style semantics define the sets of arguments that can jointly be accepted (*extensions*). A σ-extension refers to an extension under semantics σ. We consider admissible sets and preferred, complete, grounded and stable semantics with the following functions respectively adm, prf, com, grd, stb.

Definition 2. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is* conflict-free *(in F), if there are no $a, b \in S$, such that $(a, b) \in R$. The collection of sets which are conflict-free is denoted by $\mathit{cf}(F)$. For $S \in \mathit{cf}(F)$, it holds that:*

- *$S \in \mathit{adm}(F)$, if each $a \in S$ is defended by S;*
- *$S \in \mathit{prf}(F)$, if $S \in \mathit{adm}(F)$ and for each $T \in \mathit{adm}(F)$, $S \not\subset T$;*
- *$S \in \mathit{com}(F)$, if $S \in \mathit{adm}(F)$ and for each $a \in A$ defended by S it holds that $a \in S$;*
- *$S \in \mathit{grd}(F)$, if $S \in \mathit{com}(F)$ and for each $T \in \mathit{com}(F)$, $T \not\subset S$;*
- *$S \in \mathit{stb}(F)$, if for each $a \in A \setminus S$, S attacks a.*

Example 2. *The extensions of F_e under the preferred, complete and grounded semantics are: $\mathit{prf}(F) = \{\{a, c\}, \{a, d\}\}$; $\mathit{com}(F) = \{\{a\}, \{a, c\}, \{a, d\}\}$; $\mathit{grd}(F) = \{a\}$; $\mathit{stb}(F) = \{\{a, c\}, \{a, d\}\}$.*

Typical problems of interest for abstract argumentation semantics are as follows.

Definition 3. *Given an AF $F = (A, R)$, a semantics σ and some argument $a \in A$:*

- Enumeration Enum_σ*: construct all extensions prescribed by σ*
- Credulous acceptance Cred_σ*: decide if a is contained in at least one σ-extension*
- Sceptical acceptance Scept_σ*: decide if a is contained in all σ-extensions*

Example 3. *Under the preferred semantics, only argument a is sceptically accepted and arguments a, c and d are credulously accepted in F_e.*

The problem of deciding the acceptability of arguments is well-studied [Dunne and Wooldridge, 2009]. Whereas $\mathsf{Cred}_{\mathsf{grd}}$, $\mathsf{Scept}_{\mathsf{grd}}$ and $\mathsf{Scept}_{\mathsf{com}}$ can be solved in polynomial time, all other problems considered here are shown to be to NP-complete or surpassing.

3 Problem Setup

In order to solve argument acceptability problems with a GNN we pose them as a classification problem. Consider an AF $F = (A, R)$ and a semantics σ. Our goal is to approximate a function f_σ mapping the input F to a binary labelling $f_\sigma(F)$ denoting the acceptability of all arguments in A under semantics σ. The function is approximated by producing a value for each argument in the interval $[0, 1]$ - representing the likelihood whether an argument can be accepted - which is rounded to produce a binary answer (accept or reject).

4 Model

We introduce our *argumentation graph neural network* (AGNN) model. An AGNN maps an AF to a graph representation and assigns a multidimensional embedding to each node. These embeddings are then iteratively updated by performing a number of message passing steps. At each iteration nodes broadcast their embeddings by exchanging messages with their neighbours and subsequently update their embedding based on the incoming messages. After each iteration those embeddings can be read out to produce the predicted likelihood of the respective argument being accepted.

More formally, G is an AF graph representation in which arguments are nodes and attacks are directed edges. Each node i is assigned an embedding, denoted by v_i^t at step t. The node embedding is initialised by a learned embedding x_i such that $v_i^0 = x_i$. Each message passing iteration the embeddings are updated according to:

$$m_i^{t+1} = \sum_{j \in N^s(i)} M^s(v_i^t, v_j^t) + \sum_{k \in N^t(i)} M^t(v_i^t, v_k^t) \tag{1}$$

$$(v_i^{t+1}, h_i^{t+1}) = U(h_i^t, m_i^{t+1}, x_i) \tag{2}$$

where $N^s(i)$ and $N^t(i)$ denote all nodes which have a connection with node i and for which i is the source or target node respectively. The message functions M^s and M^t are Multilayer perceptrons (MLPs) which learn to compute a message to send along edges based on the embeddings of the

Characteristic	grd	prf	stb	com
Extensions per AF	1.0	2.1	1.6	6.3
Arguments per extension	4.7	9.5	11.8	8.0
Scept. accepted arguments per AF	4.8	5.9	5.8	4.8
Cred. accepted arguments per AF	4.8	8.0	7.9	8.0

Table 1: AF characteristics averaged over all AFs the test dataset.

nodes it connects. M^s computes a message from the source node to the target node and M^t vice versa. Messages from all neighbours are subsequently summed to form the incoming message m_i^t. Aggregating messages into a single incoming message allows the model to handle graphs of arbitrary size. For nodes which do not have any incoming edges m_i^t is filled with zeros. The update function U is a Recurrent Neural Network (RNN) which learns how to update a node given the incoming message and the node's input feature, where h_i^t is the RNNs hidden state. By updating the node embeddings recurrently while also accounting for the input features, AGNN is able to iteratively refine embeddings without forgetting any potentially relevant information.

After each iteration the embeddings can be read out with the readout function R. R is an MLP that learns to map a node's embedding v_i^t to a logit probability $o_{v_i}^t = R(v_i^t)$ representing the likelihood of the respective argument being accepted. This logit probability can subsequently be converted to a likelihood in the interval $[0, 1]$ using a sigmoid function.

The message and update functions form the core of the AGNN model. Together, the functions yield a neural message passing algorithm whose parameters can be optimised. The readout function serves as a mapping between the multidimensional embeddings and the output values. In terms of argumentation AGNN learns how to initialise arguments with an embedding; recurrently update these embeddings by exchanging messages between arguments over the attack relations; and map the argument embeddings to a likelihood of that argument being accepted.

5 Experimental Setup

5.1 Data

We generate a variety of challenging argumentation frameworks by sampling from the following AF generators from the International Competition on Computational Models of Argumentation [Gaggl *et al.*, 2020]: *AFBenchGen2*, *AFGen Benchmark Generator*, *GroundedGenerator*, *SccGenerator*, *StableGenerator*. To avoid duplicates, each AF is checked for isomorphism with *Nauty* [McKay and Piperno, 2014]. Ground-truth labels are determined based on extensions obtained with the sound and complete *μ-toksia* solver [Niskanen and Järvisalo, 2019]. We generate a test and validation dataset of size 1000 with AFs containing $|A| = 25$ arguments, and a training dataset of a million AFs where the number of arguments per AF is sampled randomly between $5 \leq |A| \leq 25$ (to accelerate the learning). Table 1 shows characteristics of the AFs in the test dataset under different semantics.

		Scept$_\sigma$				Cred$_\sigma$			
Metric	Model	grd	prf	stb	com	grd	prf	stb	com
MCC	GCN	0.17	0.18	0.20	0.16	0.17	0.17	0.39	0.36
	FM2	0.64	0.54	0.55	0.64	0.63	0.57	0.55	0.57
	AGNN	**1.00**	**0.997**	**0.997**	**1.00**	**1.00**	**0.998**	**0.998**	**0.999**
MAE	AGNN	3e^{-8}	5e^{-4}	9e^{-4}	3e^{-8}	3e^{-8}	6e^{-4}	4e^{-4}	3e^{-4}

Table 2: Argument acceptance results on the test dataset.

5.2 Training

We instantiate the AGNN model with one hidden layer and a rectified linear unit for non-linearity for the MLPs M^s, M^t and R, a Long Short-Term Memory [Hochreiter and Schmidhuber, 1997] for U and a shared random embedding x_i for all nodes. The dimensions of the embedding and all hidden neural layers are $d = 128$. The model is run for $\mathcal{T} = 32$ message passing steps. We train our model in batches containing 50 graphs (approximately 750 nodes) using the AdamW optimiser [Loshchilov and Hutter, 2019] with a cosine cyclical learning rate [Smith, 2017] between 2e^{-4} and 1e^{-7}, ℓ_2 regularisation of 1e^{-9} and clip the gradients by global norm with a 0.5 clipping ratio [Pascanu *et al.*, 2013]. We train the model by minimising the binary cross entropy loss between the predicted likelihood and the ground-truth binary label. We minimise the loss at every message passing step (rather than only on the final step) since this encourages the model to learn a convergent message passing algorithm while also mitigating the vanishing gradient problem [Palm *et al.*, 2018].

6 Results

We train AGGN on the AFs in the training dataset for each acceptance problem and semantics described in Section 2. We use the Matthews Correlation Coefficient (MCC) to evaluate the binary classification performance on the AFs in the test dataset. Since accepted and rejected arguments are equally important but unequally distributed in the dataset (see Table 1), MCC provides a more balanced metric compared to accuracy or the F1-score [Powers, 2011]. We compare our approach to a graph convolutional network (GCN) [Kipf and Welling, 2017] baseline and our own implementation of the FM2 model of Kuhlmann and Thimm [2019]. Both are single forward pass classifiers, where FM2 is a GCN with the number of incoming and outgoing attacks per argument added as input features. Table 2 reports the MCC scores for all models. AGNN performs considerably better than GCN or FM2, and achieves a perfect score on all problems which belong to complexity class P. On all other problems (belonging to NP or surpassing) AGNN is able to correctly predict the acceptance of arguments almost perfectly.

In addition to the classification performance of the model, we are interested in the confidence of its predictions. As a measure of prediction confidence we use the mean absolute error (MAE) between the predicted likelihoods and the binary ground-truth labels. An MAE of 0.01 implies that on average the predicted likelihood deviates 1 percentage point from the ground-truth label. A low MAE thus indicates predictions are made correctly and with high confidence. Table 2 shows that

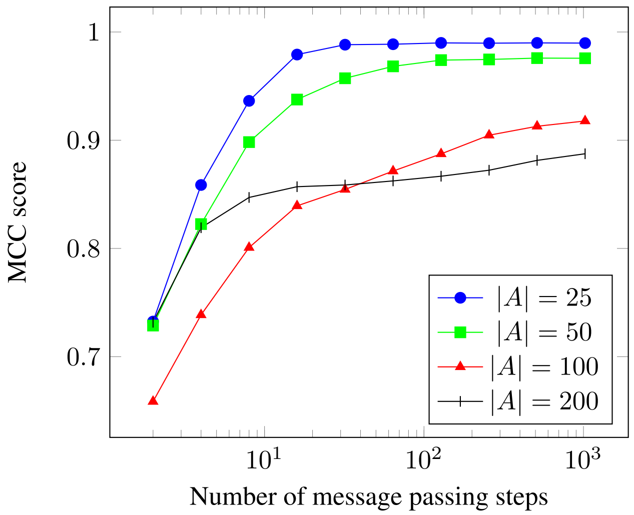

Figure 2: MCC score for $\mathsf{Cred}_{\mathsf{prf}}$ on AFs of different size as a function of the number of message passing steps $\mathcal{T}$. The performance for the first few message passing steps mainly reflects how well AGNN is able to anticipate the status of arguments before convergence. Because AFs are randomly generated the performance during this anticipation phase may vary between datasets, hence the crossing lines for $|A| = 100$ and $|A| = 200$.

predictions are overall made with high confidence. Most notably AGNN predictions deviate only $3e^{-6}$ percentage points from the true label on all problems for which it achieved a perfect classification score.

6.1 Scaling

Even though AGNN is trained on AFs of size $5 \leq |A| \leq 25$, it is able to determine acceptability in much larger AFs. In order to test how well the trained model scales to larger instances we generate extra test datasets for each $|A| \in \{50, 100, 200\}$ with 1000 AFs containing $|A|$ arguments. Figure 2 illustrates the MCC scores for predicting the credulous acceptance under the preferred semantics on different sized AFs as a function of the number of message passing steps $\mathcal{T}$. The figure shows AGNN continues to improve its predictions on large AFs by running for more iterations. Notably the performance on $|A| = 200$ AFs still improves after hundreds of iterations (which is not surprising considering that those AFs on average contain 5e^3 attacks and up to 9e^5 extensions). This indicates that, while only being trained to perform 32 message passing steps, AGNN has learned some general and convergent message passing procedure which scales to larger AFs by performing more iterations.

AGNN exhibits similar behaviour on all problems of the same complexity as $\mathsf{Cred}_{\mathsf{prf}}$. On all problems belonging to complexity class P, AGNN is able to correctly classify all arguments in AFs with $|A| = 200$ when run for 32 message passing iterations.

7 Analysing AGNN Behaviour

The AGNN model learns a message passing algorithm which enables it to predict the acceptance status of arguments in an AF. The process of iteratively updating arguments by exchanging messages can be understood as performing a sequence of relational inferences between connected arguments. Since the computations underlying those inferences are learned by neural networks, it is hard to interpret 'how' those inferences enable the model to predict acceptance.

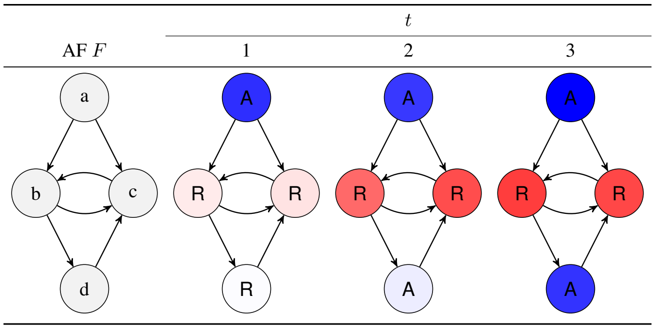

Figure 3: The acceptance predictions AGNN makes after the first three message passing iterations on the AF $F = (\{a, b, c, d\}, \{(a, b), (a, c), (b, c), (b, d), (c, b), (d, c)\}$ with respect to the grounded semantics. The label and colour of each arguments denote whether the argument is predicted to be A accepted or R rejected where a darker colour indicates a higher confidence prediction. At $t = 1$ argument a converges to *accept* while the other arguments anticipate *reject* with a confidence that positively correlates with the amount of incoming attacks. At $t = 2$ arguments b and c converge to *reject* while d adjusts its anticipation to *accept* since all its neighbours anticipated *reject*. At $t = 3$ d converges to *accept* after which the model stops evolving.

We inspect the outputs of each iteration in order to infer how the model works towards a solution. AGNN exhibits similar behaviour on all AFs in the test dataset. Arguments are initialised with a low confidence prediction. At each iteration the likelihoods change based on the incoming messages, until arguments 'decide' on their status by *converging* to a high confidence likelihood close to 0 or 1. Generally, the convergence of an argument directly affects the prediction of adjacent arguments in the next iteration. As information is exchanged between arguments, convergence propagates through the graph until the model stops evolving and the likelihoods stay more or less constant.

To gain a better understanding of this behaviour we focus on acceptance under the grounded semantics. As shown in Table 2 AGNN is able to correctly predict the acceptance of all arguments with extremely high confidence. We inspect how arguments in an AF converge over consecutive iterations (as shown in Figure 3) and observe three consistent patterns:

1. unattacked arguments converge to *accept*;
2. any argument attacked by an argument which is converged to *accept*, converges to *reject*;
3. any argument which is only attacked by arguments which are converged to *reject*, converges to *accept*.

Any argument which is not affected by these procedures converges to *reject* over the course of multiple iterations. Interestingly, each procedural pattern seems to encode some principle of the grounded semantics. Pattern 1 corresponds

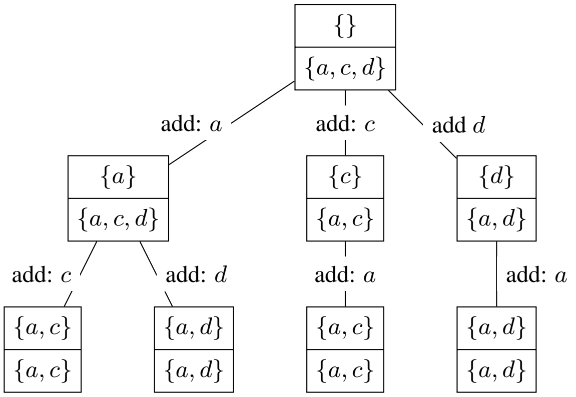

Figure 4: The search tree for enumerating the preferred extensions of the AF F_e. Each tree node illustrates a set S in the top half and the set of arguments which are constructively accepted w.r.t. S in the bottom half. At each step down the tree a constructively accepted argument is included in S until S becomes an extension (i.e. no constructively accepted argument can extend S any further).

to the notion that the defence of arguments included in the grounded extension is 'rooted' in unattacked arguments [Baroni *et al.*, 2011]; pattern 2 corresponds with the principle of conflict-freeness; and pattern 3 corresponds with the principle of defence. Since AGNN exhibits these patterns with extremely high confidence predictions (MAE of $7e^{-7}$) on every AF, it seems the model has learned to encode these principles as procedural rules into its message passing algorithm. The procedural rules also correspond with those used in a well-established symbolic labelling algorithm which can be used to find the grounded extension [Modgil and Caminada, 2009]. This algorithm applies the same principles as described in the three observed patterns. It seems AGNN has learned to encode the principles of the grounded semantics and applies these as procedural rules to determine which arguments are contained in the grounded extension.

We also observe that AGNN tries to anticipate the acceptance status of arguments in an opportunistic fashion. Arguments which have not yet converged try to anticipate their status based on information about their neighbourhood. At the first message passing step, only unattacked have enough information to converge. For all other arguments we observe a negative correlation between the degree of incoming attacks and the predicted likelihood of being accepted. Statistically seen, the more incoming attacks an argument has, the higher the chance that it is not defended against one of those attacks[1]. It seems that AGNN has learned to infer the amount of incoming attacks from the incoming message and uses this information to anticipate the predicted likelihoods. Additionally, anticipating arguments affect unconverged neighbouring arguments according to the same procedure as mentioned above. An argument attacked by arguments which anticipate *reject* will for instance anticipate *accept* or lower its confidence in anticipating *reject*.

8 Enumerating Extensions

Motivated by AGNN's ability to predict argument acceptance almost perfectly we expand our scope to the extension enumeration problem. In Section 7 we showed that AGNN is able to predict the acceptance of arguments under grounded semantics by learning a procedure to enumerate the grounded extension. Since an AF always has one grounded extension, there is a one-to-one mapping between enumerating the extension and deciding acceptance. It seems plausible that AGNN is able to learn a similar procedure under the preferred, stable and complete semantics. However, under these semantics an argument can be contained in multiple extensions and as AGNN can only output a single value per argument there is no straightforward way to directly use AGNN to enumerate all extensions.

To facilitate enumeration under multi-extension semantics we pose enumeration as a search problem and use AGNN to guide a basic search. Starting from an empty set of arguments S we construct a search tree by incrementally adding arguments to S that *extend* S into becoming an extension. When no argument argument can extend S any further we backtrack, select a new argument to extend S and continue the search. Finding extensions with this procedure requires iteratively solving which arguments can extend S into becoming an extension and verifying when S becomes an extension. To address these problems with AGNN we propose the *constructive acceptance* task Constr_σ.

Definition 4. *Given an AF $F = (A, R)$, a semantics σ, a set of arguments $S \subseteq A$ and an argument $a \in A$, a is said to be* constructively accepted *w.r.t. S if $S \cup \{a\} \subseteq \bigcup_{\mathcal{E} \in \sigma(F)} \mathcal{E}$*

An argument can only be constructively accepted w.r.t. a set which is subset equal to an extension. Given such a set S, an argument a is constructively accepted if it is either contained in S or if adding a to S yields a larger set which is also subset equal to an extension.

Example 4. *Given the set of arguments $S = \{a\}$ in F_e, arguments a, c and d are constructively accepted w.r.t. S under preferred, complete and stable semantics while only argument a is constructively accepted under grounded semantics.*

Consider AF $F = (A, R)$ and semantics σ. Starting from the empty set S we extend S into an extension by recursively computing which arguments are constructively accepted w.r.t. S and adding one of these arguments to S. We use AGNN to approximate the function f_σ mapping F and S to a binary labelling $f_\sigma(F, S)$ indicating for each argument in A if it is constructively accepted w.r.t. S. We inform AGNN which arguments are currently in S by initialising the corresponding nodes with a separate embedding x_i and we round the computed likelihoods for each argument to a binary answer.

Each time a constructively accepted argument is added, S is extended into a larger subset of an extension until at some point it becomes equal to an extension. Verifying when S becomes equal to an extension is straightforward under the grounded, preferred and stable semantics. Under these semantics no extension can be a subset of another extension. Therefore S is an extension when all constructively accepted arguments are included in S and no argument can extend

[1] This encodes the ideas behind ranking-based semantics, in which the numbers of attackers and defenders are used to rank arguments [Bonzon *et al.*, 2016].

	Enum_σ			
Metric	grd	prf	stb	com
Precision	1.00	0.999	1.00	1.00
Recall	1.00	0.998	0.999	0.41

Table 3: Extension enumeration results on the test dataset.

it any further. Those extensions are thus found in the leaf nodes of the search tree (as shown in Figure 4). Under the complete semantics this principle does not hold for all extensions. Since a complete extension can also be a subset of another complete extension, exhaustively extending S until it becomes an extension will find some, but not all extensions.

Since AGNN provides an approximation, somewhere in the tree search an argument a might falsely be labelled as constructively accepted w.r.t. S, yielding the *illegal* set $S \cup \{a\}$. An illegal set is not subset equal to any extension and therefore cannot be extended into a extension. Due to the branching nature of a tree search, a single mislabelled argument early in the search procedure may spawn many illegal sets thereby increasing the risk of an incorrectly enumerated extension. To mitigate this risk while constructing the search tree we stop extending any set S when AGNN's output indicates that S contains a constructively rejected argument (since as long as S is subset equal to an extension all argument in S should be constructively accepted by Definition 4).

8.1 Experimental Setup and Results

We alter the training dataset described in Section 5.1 to supervise AGNN in learning to predict which arguments are constructively accepted w.r.t. a set of arguments. For each AF we generate a set of arguments S which is subset equal to a randomly selected extension to serve as input feature. The ground-truth labels are determined by taking the union of all extensions which contain S and label each argument as *accepted*. To train AGNN in recognising illegal sets we also generate a set of arguments which is not subset equal to any extension and set the ground-truth label of each argument to *reject*. We train AGNN with the same parameters as described in Section 5.2.

For each AF in the test dataset we use AGNN to construct a search tree and return the sets found in the leaf nodes. Table 3 shows AGNN is able to enumerate extensions almost perfectly under most semantics. As anticipated the recall under the complete semantics is relatively low since the search procedure cannot find extensions which are a subset of another extension. However when we include a verification algorithm [Besnard and Doutre, 2004] to enable the identification of such extensions, the recall increases to 0.91. This indicates AGNN has indeed learned the principles of constructing complete extensions but many are not identified as such due to the nature of the search procedure.

9 Discussion

9.1 Related Work

Existing research on (deep) learning based approaches to argumentation focus mainly on argument mining – that is, extracting arguments or attacks from natural language text [Cocarascu and Toni, 2017] – instead of solving acceptability problems. The exception is recent work by Kuhlmann and Thimm [2019], who carried out a feasibility study on the use of a graph convolutional neural network to approximate the credulous acceptance of arguments under the preferred extension. The proposed FM2 model operates as a conventional single forward pass classifier where the number of incoming and outgoing attacks is added to each argument as input features. Thus, FM2 learns to find a correlation between the input features of an argument's neighbourhood and the likelihood of being accepted. In contrast, AGNN learns to perform a sequence of relational inferences which enable it to approximate the acceptance of an argument solely based on the attack structure of an AF. This more general approach greatly outperforms FM2's local focus (see Table 2).

From a machine learning perspective our model is close to Palm et al. [2018] and Gilmer et al. [2017]. Both describe GNNs that learn neural message passing algorithms on problems from symbolic domains. Our model differs since we employ two message functions in order to distinguish between messages sent from attack source to target or vice versa. In addition we show how GNNs can be used to guide a basic search on problems for which multiple solutions exist.

9.2 Conclusion and Future Work

We have presented a learning-based approach to determining acceptance of arguments under abstract argumentation semantics, proposing AGNN, which learns a message-passing algorithm to predict the likelihood of an argument being accepted. AGNN can almost perfectly predict the acceptability under different semantics and scales well for larger argumentation frameworks. Furthermore, AGNN can also enumerate all extensions under different semantics very well - for multi-extension semantics, AGNN is used to extend a set of arguments such that it becomes an extension. Analysis of AGNN's behaviour shows that it learns to adhere to basic principles of (ranked) argument semantics as identified in the literature [Baroni *et al.*, 2011; Bonzon *et al.*, 2016], and behaves similarly to a well-known symbolic labelling algorithm for grounded semantics [Modgil and Caminada, 2009].

Although AGNN does not provide the same theoretical guarantees as a symbolic algorithm, the appeal of a learning-based approach is that it generalises to different problems without needing the expert knowledge of human algorithm designers [Li *et al.*, 2018]. AGNN is a single architecture that can solve different argumentation problems (Cred, Scept, Constr) for different semantics (grd, prf, stb, com) and on AFs larger than seen during training in constant time, simply by running for more iterations. Additionally, by solving the constructive acceptance problem, AGNN can guide a basic tree search enumerating extensions with a polynomial delay.

For future work, we aim to look at employing AGNN for dynamic argumentation [Doutre and Mailly, 2018], looking at whether AGNN can learn, for example, which arguments or attacks should be added or removed to enforce a certain argument's acceptability.

References

[Atkinson *et al.*, 2017] K. Atkinson, P. Baroni, M. Giacomin, A. Hunter, H. Prakken, C. Reed, G. R. Simari, M. Thimm, and S. Villata. Towards artificial argumentation. *AI Magazine*, 38(3):25–36, 2017.

[Baroni *et al.*, 2011] P. Baroni, M. Caminada, and M. Giacomin. An introduction to argumentation semantics. *Knowledge Engineering Review*, 26(4):365–410, 2011.

[Battaglia *et al.*, 2018] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv e-prints*, arXiv:1806.01261, 2018.

[Besnard and Doutre, 2004] P. Besnard and S. Doutre. Checking the acceptability of a set of arguments. In *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR'04)*, pages 59–64, 2004.

[Bonzon *et al.*, 2016] E. Bonzon, J. Delobelle, S. Konieczny, and N. Maudet. A comparative study of ranking-based semantics for abstract argumentation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 914—-920, 2016.

[Charwat *et al.*, 2015] G. Charwat, W. Dvorák, S. A. Gaggl, J. P. Wallner, and S. Woltran. Methods for solving reasoning problems in abstract argumentation - A survey. *Artificial Intelligence*, 220:28–63, 2015.

[Cocarascu and Toni, 2017] O. Cocarascu and F. Toni. Identifying attack and support argumentative relations using deep learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1374–1379, 2017.

[d'Avila Garcez *et al.*, 2015] A. S. d'Avila Garcez, T. R. Besold, L. D. Raedt, P. Földiák, P. Hitzler, T. Icard, K. Kühnberger, L. C. Lamb, R. Miikkulainen, and D. L. Silver. Neural-symbolic learning and reasoning: Contributions and challenges. In *Proceedings of the 2015 AAAI Spring Symposia*, Palo Alto, 2015. Stanford University.

[Doutre and Mailly, 2018] S. Doutre and J. Mailly. Constraints and changes: A survey of abstract argumentation dynamics. *Argument & Computation*, 9(3):223–248, 2018.

[Dung, 1995] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[Dunne and Wooldridge, 2009] P. E. Dunne and M. J. Wooldridge. Complexity of abstract argumentation. In G. R. Simari and I. Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 85–104. Springer, 2009.

[Gaggl *et al.*, 2020] S. A. Gaggl, T. Linsbichler, M. Maratea, and S. Woltran. Design and results of the second international competition on computational models of argumentation. *Artificial Intelligence*, 279:103193, 2020.

[Gilmer *et al.*, 2017] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1263–1272, 2017.

[Hochreiter and Schmidhuber, 1997] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[Kipf and Welling, 2017] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations)*, 2017.

[Kuhlmann and Thimm, 2019] I. Kuhlmann and M. Thimm. Using graph convolutional networks for approximate reasoning with abstract argumentation frameworks: A feasibility study. In *Proceedings of the 13th international conference on Scalable Uncertainty Management (SUM)*, pages 24–37, 2019.

[Li *et al.*, 2018] Z. Li, Q. Chen, and V. Koltun. Combinatorial optimization with graph convolutional networks and guided tree search. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 537–546, 2018.

[Loshchilov and Hutter, 2019] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations)*, 2019.

[McKay and Piperno, 2014] B. D. McKay and A. Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60:94–112, 2014.

[Modgil and Caminada, 2009] S. Modgil and M. Caminada. Proof theories and algorithms for abstract argumentation frameworks. In I. Rahwan and G. R. Simari, editors, *Argumentation in Artificial Intelligence*, pages 105–129. Springer, 2009.

[Niskanen and Järvisalo, 2019] A. Niskanen and M. Järvisalo. μ-toksia: Sat-based solver for static and dynamic argumentation frameworks. https://bitbucket.org/andreasniskanen/mu-toksia, 2019.

[Palm *et al.*, 2018] R. B. Palm, U. Paquet, and O. Winther. Recurrent relational networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3372–3382, 2018.

[Pascanu *et al.*, 2013] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1310–1318, 2013.

[Powers, 2011] D. Powers. Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machice Learning Technology*, 2:2229–3981, 01 2011.

[Selsam *et al.*, 2019] D. Selsam, M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill. Learning a SAT solver from single-bit supervision. In *7th International Conference on Learning Representations (ICLR)*, 2019.

[Smith, 2017] L. N. Smith. Cyclical learning rates for training neural networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.