Universität Stuttgart
Institute for Parallel and Distributed Systems
Department Analytic Computing

29.06.2021

Explanation-based Learning with Feedforward Neural Networks

Proposal

By Jan Gemander

Supervisors: Prof. Steffen Staab, Zihao Wang

1. Introduction:

The idea of this thesis is to adapt Možina et al.'s method [2] of exploiting experts' arguments to neural networks. This is done by incorporating the method of Ross et al. [7] to include an explanatory loss, which penalizes attention on the wrong features. More specifically the novelty of this work is to focus on both positively and negatively influencing features depending on experts' arguments. Contributions are computed using the shapley value and approximations thereof, with the goal in mind to achieve improved training performance as well as improving the focus on the correct features. Additionally, by concentrating the neural network on features in the experts' arguments, explanations generated for predictions of our network should be more similar to those of experts without relying on unfamiliar dependencies.

2. Related Work:

Argument based machine learning (ABML) is about using arguments supporting and attacking an assertion for correct classification. Možina et al. [2] used in 2007 a covering algorithm on annotated classification data such as [credit: approved, because: rich, despite: not paying regularly] to mine rules. These had to support positive arguments while omitting negative arguments. Explanations are then immediately given by the predicting rule.

Attention is often used as a proxy for reasoning, even more so for text and image classification tasks. The relevance of such attention is disputed with both papers, "Attention is not Explanation" [4] and "Attention is not not explanation" [5] being released in 2019. Arous et al. [6] learned a model that incorporates attention for text classification using a Bayesian inference framework. The attention itself was derived from human annotators, including a learned reliability factor of the workers to counteract bad annotations.

The field of explainable interactive learning (XIL) focuses on interacting with the explanations of a predictor model, to prevent cases where the model predicts the correct result because of the wrong reasons. Ross et al. [7] introduced an additional explanatory loss factor which should restrict the networks explanations to the right reasons. The loss factor penalizes the attention on wrong reasons, by summing up their gradients. Stammer et al. [3] expand the explanatory loss term to include explanations of both a concept and a reasoning module attempting to teach the network the correct concept. The reasoning module estimates the importance of symbolic representations using the integrated gradient method, while the concept explanations are an attention mapping restricted by importance of the respective symbolic representation.

ELIXIR by Ghazimatin et al. [8] is a graph recommender system that that features explanation for recommendations that the user can interact with. More specifically a feedback matrix is stored for

every user that incorporates how much a user likes <item, explanation> combination pairs to create further recommendations.

3. Foundations:

In this section we want to introduce foundations from other research that this paper heavily relies on.

3.1 Argument based machine learning (ABML)

The rule mining in [2], which we want to adapt to neural networks, uses an argumented example AE for every datapoint:

Def. 1. An argumented example AE is a four-tuple (x, y^*, A^+, A^-), comprising an example (x, y^*), as well as a set of positive arguments A^+ and a set of negative arguments A^- with $A^+ \cap A^- = \emptyset$. A^+ comprises reasons that are important for the classification of y^*. A^- comprises reasons which "speak against the class value" [2].

Here reasons are a conjunction of reasons $r_1, \dots, r_n$. Each of these reasons refers to the relationship $R \in [=, <, >]$ of a feature x_i to a value $z_i : x_i R z_i$, where z_i is of the same domain as x_i.

For example, consider the decision of a bank to give a loan to a person. An expert would note down the reasons for the arguments, as to why the outcome was positive, such as:

Because A^+: $rich = true,\ monthly\ income > 1000$
Despite A^-: $pays\ on\ time = false$

Features can include other attributes such as gender, age, or hair color. However, arguments should only contain reasons with features that have an impact on the class value. Reasons in A^+ are then used as a starting point for the rule mining algorithm, while reasons in A^- must be omitted.

Rules consist of a conjunction of selectors that imply a target class. For simplification, a selector is of the same composition as a reason. The rule mining algorithm starts by evaluating the arguments as if they were rules and proceeds to refine them. An AE is then covered by a rule if all selectors are true for the example, at least one reason $r_i \in A^+$ is included in the selectors, and no reasons $r_j \in A^-$ is a part of the selectors. Consider the following rules for the above credit loan example:

Rule 1: IF $rich = true \wedge pays\ on\ time = false$ THEN $loan\ granted = true$
Rule 2: IF $gender = female \wedge age > 20$ THEN $loan\ granted = true$
Rule 3: IF $monthly\ income > 1000$ THEN $loan\ granted = true$

Only Rule 3 covers the above example. Rule 1 is disregarded because it includes a negative reason. Rule 2 cannot cover the example because it does not include either of the positive reasons.

3.2 Shapley value

ABML prioritizes reasons in arguments for the generation of rules. Later, we want to adapt their approach to neural networks by focusing our network on features in arguments. To ensure that the correct features are responsible for the decision of a network, we want to maximize their contributions to the prediction. Therefore, we require a method to compute the contributions of features. A natural choice for this task is the shapley value from game theory, where players $N = \{p_1, \dots, p_n\}$ cooperate in a coalition S for a total payout $v(S)$.

Def. 2. The Shapley value $sh_i(v)$ for a player p_i for a game (N, v) is defined by the players average contribution to the payout:

$$sh_i(v) = \frac{1}{n!} \sum_{S \subseteq N \setminus \{p_i\}} |S|! \, (n - |S| - 1) \, ! \left(v(S \cup p_i) - v(S) \right) \tag{1.1}$$

This function sums up the contributions of a player p_i when added to a coalition S for all possible subsets of N. The contribution is multiplied by the possible permutations of the current ($|S|!$) and remaining ($(n - |S| - 1)!$) players. Finally, the function averages over all possible permutations ($\frac{1}{n!}$).

For a three-player game with $v(\emptyset) = v(1) = v(2) = v(3) = 0, v(1,2) = 1, v(1,3) = 2, v(2,3) = 3$ and $v(1,2,3) = 6$ the shapley value for player 1 would be computed as follows:

$$\begin{aligned} sh_1(v) &= \frac{1}{3!} * (2 * \left(v(1) - v(\emptyset) \right) + \left(v(1,2) - v(2) \right) + \left(v(1,3) - v(3) \right) + 2 * (v(1,2,3) - v(2,3)) \\ &= \frac{1}{6} * (2 * (0 - 0) + (1 - 0) + (2 - 0) + 2 * (6 - 3) \\ &= 1.5 \end{aligned}$$

The shaplay value vector for all players is then given by $sh(v) = \left(sh_1(v), \dots, sh_n(v) \right)^T$. By replacing the value function with a network $f(x)$ and viewing features $x_1, \dots, x_n$ as the players we can use the shapley value to estimate feature's contributions to predictions in machine learning. The shapley value computation for the contribution of feature x_i to a network's prediction $f(x)$ with input $x = (x_1, \dots, x_n)$ and set of all features $N = \{x_1, \dots, x_n\}$ would be adapted as follows:

$$sh_i(f, x) = \frac{1}{n!} \sum_{S \subseteq N \setminus \{x_i\}} |S|! \, (n - |S| - 1) \, ! \left(f(S \cup x_i) - f(S) \right) \tag{1.2}$$

A network with multiple output values $f(x) = (y_1, \dots, y_m)^T$ has shapley values of features for every output respectively. For a network with a single input $x = x_1, \dots, x_n$ and output $f(x) = (y_1, \dots, y_m)^T$ the contribution of a feature x_i to output y_j is given by the shapley value $sh_{i,j}(f, x)$. The vector specifying the contributions of all inputs to one output y_j is given by $sh_{*,j}(f, x)$.

For example, for a network with an input $x = (x_1, \dots, x_{10})$ of length 10 and an output of length 3 $f(x) = (y_1, y_2, y_3)$ the contribution of feature x_4 to the output y_3 is given by $sh_{4,3}(f, x)$. The matrix for all possible contributions in this example would be of size 10×3, where each of the three columns describes the contributions of all features to the respective output independently. More precisely the second column describes the contributions of every feature with respect to y_2.

Intuitively these contribution values explain the results of a prediction similarly to the arguments of experts, and as such, can be used to generate explanations for a user with no prior knowledge.

Def. 3. Given a predictor f, a data point $d = (x, y)$ and the set of all possible explanations E, an explanator function ϵ is the mapping ϵ: $(X^n \to Y) \times (X^n) \to \mathrm{E}$, taking the predictor function f and data point attributes x as input, and returning a human understandable explanation $e \in \mathrm{E}$ of the prediction, e.g. $\epsilon(f, x) = e$.

Here a human understandable explanation refers to a format that the human can make sense of (e.g. image, text) to understand the outcome of the prediction. Such an explanation may be an image marking important regions or a vector specifying how important each feature was. Abstract vector spaces such as word embeddings are an example for "not human understandable". The shapley value is one among many explanator functions.

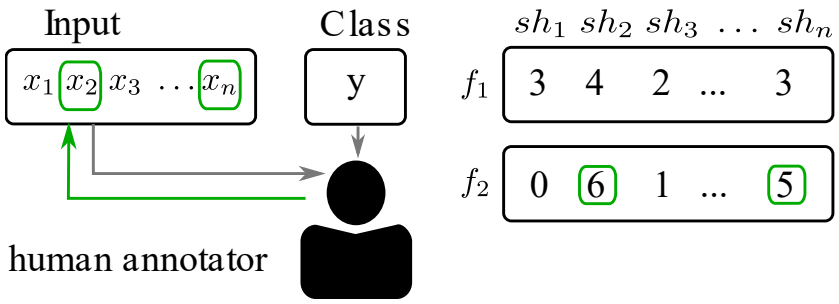

Figure 1 Annotator looking at a datapoint (grey arrow) selects features (green arrow) for positive arguments of y (green squares), f_1 is a network trained without the focus on selected features, f_2 focuses contributions (shapley values) on selected features.

4. Methodology:

This thesis attempts to adapt the method of ABML to neural networks. By including experts' arguments in neural networks, we hope to improve the focus on the correct features. For clarity we will refer to all input knowledge as arguments. Every time we refer to explanations, we mean information for the user. In this section we mainly want to describe our method of reinforcing the importance of features selected in experts' arguments, an example of this method in action can be seen in Figure 1.

4.1 Implementing contributions as loss

In a first step we attempt to implement the focus on the important features of ABML in neural networks akin to f_2 in Figure 1. These important features are selected as arguments by a human annotator for every example individually. While ABML required threshold values for reasons as a starting point for the selectors of the rules, this is not required for neural networks and selecting important features should be sufficient. We therefore redefine our arguments A to be the collection of features selected by the annotator, with positive arguments A^+ comprising features which are beneficial for the classification, and negative arguments A^- comprising features with a negative indication for the class. For the credit loan example in chapter 3.1 these sets would contain the selected features "$rich$", "$monthly\ income$" and "$pays\ on\ time$":

$$A^+ = \{rich, monthly\ income\} \text{ and } A^- = \{pays\ on\ time\}$$

Consider a network f, parameterized by its weights θ with inputs $x \in \mathbb{R}^n$ with $x = (x_1, \dots, x_n)$ and probability vector outputs $f(x) = \hat{y} = (\hat{y}_1, \dots, \hat{y}_m)^T$ responding to one hot vectors $y \in R^m$. For training we use an argumented example AE for every datapoint (if possible). We use the notation $y^* = j$ to denote that a one hot vector with $y_j = 1$ would be the correct prediction for the network.

For every training sample (x, y^*) with $x = (x_1, \dots, x_n)$ additional knowledge is given in the form of arguments, which specify the subset of important features. To ensure that our network incorporates this knowledge we can either reward using important features or penalize using others. Ross et al. [7] used the latter method in their RRR loss, by adding an additional loss factor that penalizes attention on the wrong features. We want to adapt their method, but instead use the former method by rewarding a focus on features in positive arguments. More specifically we add an argument loss factor that rewards positive contributions of features in positive arguments $x_i \in A^+$ to the correct prediction $y^* = j$. For every training example (x, y^*) with $x = (x_1, \dots, x_n)$ the contributions of features are computed on our network f individually. The contributions are given by a function $\phi_{i,j}(f, x)$, such as the shapley value equation (1.2), which estimates the contribution of a feature x_i to the prediction $\hat{y}_j$.

For an argumented example AE, with feature vector $x = (x_1, \dots, x_n)$, arguments A^+ and A^-, of class $y^* = j$ the extended loss L for the above network with weights θ would be computed as follows:

$$L_f(x, y^*) = l(y^*, f(x)) - \lambda_1 \sum_{x_i \in A^+} \phi_{i,j}(f, x) + \lambda_2 \sum_i |\theta_i| \tag{2.1}$$

Here l is the prediction error calculated using a loss function such as the cross entropy, and λ is used as weighing factors for the argument loss and weight regularization terms. In the argument loss term, we reward positive contributions of features in positive arguments by subtracting them from the total loss. This way the network should focus on the features which were selected by the annotator. We apply L1 regularization to the weights θ to encourage smaller weights and avoid overfitting. Additionally, L1 regularization leads to more sparse weights.

In a second step we want to incorporate the negative arguments A^- by adding the contributions of features $x_i \in A^-$ to the argument loss. Their positive contributions are therefore penalized, while negative contributions would be rewarded. This would turn the above equation into:

$$L_f(x, y^*) = l(y^*, f(x)) - \lambda_1 \sum_{x_i \in A^+} \phi_{i,j}(f, x) + \lambda_2 \sum_{x_i \in A^-} \phi_{i,j}(f, x) + \lambda_3 \sum_i |\theta_i| \tag{2.2}$$

A big difference in our method in comparison to ABML is the fact that all features not selected for A^+ are actively penalized. In ABML the mined rules had to cover a reason in A^+, but features not included in these reasons could also be important.

4.2 Computation of contributions

The pipeline of our method is as follows:

1. Annotator selects features for A^+/A^-.
2. Optimize network by maximizing (A^+)/minimizing (A^-) contributions of selected features using shapley value for every training example.
3. Use explanator function such as the shapley values on new datapoints classified by the trained network to generate their explanations.

Consider a training dataset D with data points $d = (x, y^*)$ with $x = (x_1, \dots, x_n)$. In the first step of our pipeline an annotator selects the corresponding features $x_i \in x$ for positive arguments A^+ and negative arguments A^- for every data point individually. The second step utilizes the shapley value in training of the network as described in chapter 4.1 to ensure that the selected features contribute the most to the prediction. The computation of feature contributions to the prediction via the shapley value is done using the equation (1.2). After training the network we can then compute the shapley value for new data points to compute their explanations. If we seek sparse explanations other explanators such as LIME or SHAP can be used here in place of the shapley value. Explanations generated on our network should be more similar to arguments selected by an annotator then explanations generated on a network without the argument loss.

Considering that the shapley value has to compute the result for every possible subset of features, it gets computationally exponentially expensive with increased feature count. Therefore, we suggest comparing it to the following approximation methods ϕ:

1. Gradient: This method was used by Ross et al. [7] as a first order explanation. It computes the value of a feature x_i using its gradient and scales it with its value:

$$\phi_i(f, x) = x_i * \frac{\partial \mathrm{f(x)}}{\partial \mathrm{x_i}} \tag{3.1}$$

2. Integrated gradient: This method was used in reasoning module by Stammer et al. [3]. In comparison with the gradient, it additionally satisfies the sensitivity axiom[1]. This is done by integrating the above gradient:

$$\phi_i(f, x) = x_i * \int_{\alpha=0}^{1} \frac{\partial f(\alpha * x)}{\partial x_i} \tag{3.2}$$

Especially the last method has been a popular variant to generate explanations in recent years. As an additional note, implementing baseline values instead of occlusion, that would be subtracted from the feature value x_i in the methods 2 and 3, usually improves the results.

5. Evaluation:

5.1 Data:

For the evaluation of the framework, we want to use varying datasets where annotators can create arguments for examples with respect to their classes.

The following two datasets were used in [2] which enables us to use them as a reference point:

1. Japanese Credit Screening Data [1] features 125 instances and can be used as described in the above example. The South German Credit Data Set[2] albeit not included in [2] features the same problem with 1.000 more instances.

2. Zoo Data Set [3] features 101 animals with 17 attributes that are classified into 7 types (mammals, reptile, etc.). Various features are correlated to different classes in only specific examples (e.g. laying eggs), which makes this one challenging.

The following datasets were not used in [2] but include features that can be used for arguments:

3. Heart Disease Data Set[4] predicts the likelihood of a heart disease featuring 303 examples with 14 main features, that can be beneficial or negative depending on their value. Similar collections can again be found on Kaggle to extend the database.

4. The Apartment for rent classified Data Set[5] features 10.000 instances of rental housing in the US with 22 features. Instead of contributions to the classification task, we would be looking at the total contribution to the cost. Other cost estimation datasets (e.g. medical cost) can be used in a similar way.

All of the above datasets have not been annotated with positive or negative Arguments. This has to be done either by hand or by applying general knowledge to them. For the first two datasets creating such arguments should be more straightforward, especially considering that arguments from [2] can be included.

[1] https://archive.ics.uci.edu/ml/datasets/Japanese+Credit+Screening
[2] https://archive.ics.uci.edu/ml/datasets/South+German+Credit
[3] http://archive.ics.uci.edu/ml/datasets/Zoo
[4] https://archive.ics.uci.edu/ml/datasets/Heart+Disease
[5] https://archive.ics.uci.edu/ml/datasets/Apartment+for+rent+classified.

5.2 Metrics:

For our network we want to evaluate both the performance of the predictions as well as the quality of their explanations (e.g. shapley vector $sh_{*,j}$ for prediction y_j). The evaluation of the predictions should be as straight forward as computing the usual F1 score and precision.

To evaluate how well our network does in focusing on the correct features, we suggest looking at explanations generated on our network. We can then compare manual explanations generated by experts as a gold standard, to shapley values computed for predictions of our network. By computing the F1 score, precision and recall for the correct classification into positive and negative arguments we can evaluate if our network focuses on the correct features.

Alternatively, it can be attempted to create a scoring function to evaluate the contributions. Our approach computes the percentage of correctly attributed contributions. We normalize the shapley vector over all features for the prediction y_j by dividing it by its L1 norm: $\overline{sh_{*,j}}(f,x) = \frac{sh_{*,j}(f,x)}{|sh_{*,j}(f,x)|_1}$. If $y^* = j \; and \; y_j = 1$, we can then sum the absolute contributions of the normalized shapley vector. For features $x_i \in A^+ \cup A^-$ the score s of explanation e with normalized shapley vector $\overline{sh_{*,j}}(f,x)$ would be computed as follows:

$$s\big(e(x,f(x))\big) = \sum_{x_i \; in \; A^+} \max\left(0, \overline{sh_{i,j}}(f,x)\right) - \sum_{x_i \in A^-} \min\left(0, \overline{sh_{i,j}}(f,x)\right) \tag{4}$$

Here we reward positive contributions of features in positive arguments and negative contributions of features in negative arguments. Through the normalization we can determine what the proportion of correctly attributed features is. The value range of this function should be between 0 (none of the contributions were made by features in arguments) and 1 (all contributions to the prediction were from features in arguments).

For example, consider a network with an input $x = (x_1, x_2, x_3)$ where an annotator selected the arguments $A^+ = \{x_1\}$ and $A^- = \{x_3\}$, then we can compute the score for a shapley value explanation as follows:

E1: $sh_{*,j}(f,x) = (4,2,-4)$ then $\overline{sh_{i,j}}(f,x) = \left(\frac{4}{|4|+|2|+|-4|}, \frac{2}{|4|+|2|+|-4|}, -\frac{4}{|4|+|2|+|-4|}\right)$

$$s(e\big(x,f(x)\big) = \max\left(0, \frac{4}{10}\right) - \min\left(0, \left(-\frac{4}{10}\right)\right) = \frac{4}{10} + \frac{4}{10} = 0.8$$

As described by the arguments, positive contributions of x_1 and negative contributions of x_3 are desired. These are 4 and -4 respectively. The total sum of absolute contributions is 10, where 80% of the contributions were done by the features in arguments.

E2: $sh_{*,j}(f,x) = (4,2,4)$ then $\overline{sh_{i,j}}(f,x) = \left(\frac{4}{|4|+|2|+|4|}, \frac{2}{|4|+|2|+|4|}, -\frac{4}{|4|+|2|+|4|}\right)$

$$s(e\big(x,f(x)\big) = \max\left(0, \frac{4}{10}\right) - \min\left(0, \left(\frac{4}{10}\right)\right) = \frac{4}{10} + 0 = 0.4$$

Once again, we want positive contributions for x_1 and negative contributions for x_3. While the contribution of x_1 remains 4, the contribution of x_3 is positive in this example and therefore no longer correct. Since the total sum of contributions is still 10, the proportion of correctly attributed contributions in this example is only 40%.

Depending on remaining time we can also have a look at how well our network does in predicting datapoints outside of the trained value range. Traditionally neural networks are good at predicting datapoints within the trained scope but are bad at extrapolating predictions outside of the trained scope. It could be interesting how our network compares, especially when looking at arguments.

5.3 Baselines:

For comparison we want to introduce a few baseline options.

First, we want to compare the developed method to the same network without the included argument loss. The reasoning module in [3] serves as another comparison. It uses a more advanced version of the loss in [7], which we are using in our method.

6. Time plan:

Assuming the thesis has a duration of 6 months ($\approx 26\ weeks$), the scheduled time plan is as follows:

- Implementation: 12 weeks
  - Core Algorithm: 4 weeks
  - Various contribution functions: 2 weeks
  - Adapting neural net to datasets: 2 weeks
  - Problem solving: 4 weeks
- Testing + Evaluation: 10 weeks starting after 8 weeks
  - Testing: 2 weeks
  - Prepare evaluation: 3 weeks
  - Conduct evaluation: 3 weeks
  - Result Analysis: 2 weeks
- Writing Thesis: 14 weeks starting after 12 weeks
  - Related Work: 1 week
  - Introduction: 2 weeks
  - Methodology: 2 weeks
  - Evaluation: 2 weeks
  - Other Topics: 2 weeks
  - Revision + Printing: 5 weeks

7. Organizational matters

Duration of work:	6 months
Candidate:	Jan Gemander
E-Mail:	st114979@stud.uni-stuttgart.de
Student number:	2949130
Primary supervisor:	Prof. Dr. Steffen Staab
Supervisor:	Mr. Zihao Wang

8. References:

[1] M. Sundararajan, A. Taly und Q. Yan, „Axiomatic Attribution for Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017.

[2] M. Mozina, J. Zabkar und I. Bratko, „Argument based machine learning," *Artif. Intell.,* Bd. 171, p. 922–937, 2007.

[3] W. Stammer, P. Schramowski und K. Kersting, „Right for the Right Concept: Revising Neuro-Symbolic Concepts by Interacting with their Explanations," *CoRR,* Bd. abs/2011.12854, 2020.

[4] S. Jain und B. C. Wallace, „Attention is not Explanation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 2019.

[5] S. Wiegreffe und Y. Pinter, „Attention is not not Explanation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 2019.

[6] I. Arous, L. Dolamic, J. Yang, A. Bhardwaj, G. Cuccu und P. Cudré-Mauroux, „MARTA: Leveraging Human Rationales for Explainable Text Classification," 2021. URL:https://exascale.info/assets/pdf/arous2021aaai.pdf

[7] A. S. Ross, M. C. Hughes und F. Doshi-Velez, „Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 2017.

[8] A. Ghazimatin, S. Pramanik, R. S. Roy und G. Weikum, „ELIXIR: Learning from User Feedback on Explanations to Improve Recommender Models," *CoRR,* Bd. abs/2102.09388, 2021.

9. Signatures

_____ _____

Jan Gemander Prof. Dr. Steffen Staab

_____

Zihao Wang

10. Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

I am aware that the thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the thesis as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future theses submitted. Further rights of reproduction and usage, however, are not granted here.

This paper was not previously presented to another examination board and has not been published.

Backnang, on June 29, 2021 _____

Jan Gemander