This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test

A Study on Confidence: an Unsupervised Multi-Agent Machine Learning Experiment

Amid Mozelli, Nima TaheriNejad, and Axel Jantsch
TU Wien, Vienna, Austria
E-mail: {amid.mozelli, nima.taherinejad, axel.jantsch}@tuwien.ac.at

***Abstract*—Computational Self-Awareness (CSA) is a growing field that has been applied to various applications, which often uses Machine Learning (ML). One of the key metrics for assessing the quality of both CSA and ML systems is *confidence*, which has been used in many applications recently. Confidence has shown a great promise in improving systems' performance, in particular regarding the reliability of operations. However, from an engineering point of view, the nature of confidence as a metric has been an open question. Understanding the nature of confidence can help the better usage of the concept and, consequently, the design of better systems. Uncovering the true nature of confidence, however, is not currently within our reach. Therefore, in this work, we take one step in that direction by designing a socially-inspired experiment to investigate the nature of confidence in the context of (self-)learning. Our experiment shows that among the two candidates discussed in the literature, *probability* is a better metric for confidence. This observation sheds light on this open question and marks an entry point for further investigating the concept of confidence as a metric in ML and CSA.**

***Index Terms*—Confidence, Computational self-awareness, Ob-**

Machine Learning, Computational Self-Awareness, and Confidence

Machine Learning (ML) has a leading role in today's technology, especially in Artificial Intelligence (AI) and smart devices. Nowadays, rarely a device can be found that does not take advantage of ML. One notable concept in this area of AI is Computational Self-Awareness (CSA) [L$^+$16], [K$^+$17], [BDE$^+$20], which often relies on ML to deliver the performance and qualities it promises. A self-aware system observes its state and behavior of the environment and reacts accordingly [D$^+$16]. CSA is inspired by sociological and psychological processes (e.g., c.f., [L$^+$15], [P$^+$15], [B$^+$19]), however, it has been applied to many engineering applications [R$^+$15], [Lan17], [T$^+$17], [S$^+$17], [AP18], [S$^+$18], [S$^+$20], [K$^+$20].

In [T$^+$16], the authors have reviewed the potential role of the comprehensive observation in self-aware systems and its aspects such as, data reliability [A$^+$17], [G$^+$17], attention [A$^+$17], [T$^+$17], history [G$^+$17], [G$^+$18], and confidence which is also our focus in this paper. As defined in [T$^+$16], confidence is a metric that indicates the reliability of algorithms and processes within the system in hand. A critical, and yet often overlooked, parameter in benchmarking ML systems and application. Many works benefited from the use of confidence in their learning processes [F$^+$18], [K$^+$18], [N$^+$18]. As we are advancing in this field, a better understanding of this concept, i.e., confidence, allows us to better comprehend problems at hand and design better solutions.

Authors are with the Institute of Computer Technology (ICT), TU Wien, Gusshausstrasse 27-29 / 384, 1040 Vienna, Austria. A. Mozelli and A. Jantsch are with the Christian Doppler Laboratory for Embedded Machine Learning.

Side-box References

[A$^+$17] Arman Anzanpour et al. Self-awareness in remote health monitoring systems using wearable electronics. In *Proceedings of (DATE)*, 2017.

[AP18] Aamir Akbar and Peter R. Lewis. Self-adaptive and self-aware mobile-cloud hybrid robotics. In *4th Int. Workshop on MCSMS*, 2018.

[B$^+$19] C. M. Barnes et al. Social action in socially situated agents. In *13th IEEE SASO*, pages 97–106, 2019.

[BDE$^+$20] K. Bellman, N. Dutt, L. Esterle, A. Herkersdorf, A. Jantsch, C. Landauer, P. R. Lewis, M. Platzner, N. TaheriNejad, and K. Tammemäe. Self-aware cyber-physical systems. *ACM Transactions on Cyber-Physical Systems*, Accepted for Publication:1–24, 2020.

[D$^+$16] Nikil Dutt et al. Toward smart embedded systems: A self-aware system-on-chip (soc) perspective. *ACM TECS*, 15(2):1–27, 2016.

[F$^+$18] Farnaz Forooghifar et al. Self-aware wearable systems in epileptic seizure detection. In *21st DSD*, pages 426–432. IEEE, 2018.

[G$^+$17] Maximilian Götzinger et al. On the design of context-aware health monitoring without a priori knowledge; an AC-motor case-study. In *30th IEEE CCECE*, pages 1–5. IEEE, 2017.

[G$^+$18] Maximilian Götzinger et al. Applicability of context-aware health monitoring to hydraulic circuits. In *44th IECON*. IEEE, 2018.

[K$^+$17] Samuel Kounev et al., editors. *Self-Aware Computing Systems*. Springer, 2017.

[K$^+$18] H. A. Kholerdi et al. Enhancement of classification of small data sets using self-awareness; an iris flower case-study. In *IEEE ISCAS*, pages 1–5, 2018.

[K$^+$20] Christian Krupitzer et al. Introduction to the special issue "applications in self-aware computing systems and their evaluation". *Computers*, 9(1):1–6, 3 2020.

[L$^+$15] Peter R. Lewis et al. Architectural aspects of self-aware and self-expressive computing systems. *IEEE Computer*, 48:62–70, 2015.

[L$^+$16] Peter R. Lewis et al., editors. *Self-aware Computing Systems: An Engineering Approach*. Springer, 2016.

[Lan17] C. Landauer. Mitigating the inevitable failure of knowledge representation. In *2017 IEEE ICAC*, pages 239–246, 2017.

[N$^+$18] Katayoun Neshatpour et al. Icnn: An iterative implementation of convolutional neural networks to enable energy and computational complexity aware dynamic approximation. In *DATE*, pages 551–556. IEEE, 2018.

[P$^+$15] J. S. Preden et al. The benefits of self-awareness and attention in fog and mist computing. *Computer*, 48(7):37–45, 2015.

[R$^+$15] B. Rinner et al. Self-aware and self-expressive camera networks. *Computer*, 48(7):21–28, 2015.

[S$^+$17] L. C. Siafara et al. Samba: A self-aware health monitoring architecture for distributed industrial systems. In *43rd IECON*, pages 3512–3517, 2017.

[S$^+$18] Lydia Chaido Siafara et al. Samba – an architecture for adaptive cognitive control of distributed cyber-physical production systems based on its self-awareness. *e & i Elektrotechnik und Informationstechnik*, 135(3):270–277, Jun 2018.

[S$^+$20] E. Shamsa et al. User-centric resource management for embedded multi-core processors. In *Proc. of 33rd IEEE (VLSID)*, pages 1–6, 2020.

[T$^+$16] N. TaheriNejad et al. Comprehensive observation and its role in self-awareness; an emotion recognition system example. In *(FedCSIS)*, 2016.

[T$^+$17] N. TaheriNejad et al. Self-aware sensing and attention-based data collection in multi-processor system-on-chips. In *15th IEEE NEWCAS*, pages 81–84, 2017.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test

I. Introduction

In this paper, we try to evaluate and display the effect of various methods of assessing confidence in practice in a socially inspired learning experiment. In these experiments, an unsupervised community of learning agents learns to solve a mapping problem based on a communal (social) decision process. More specifically, in our experiments, we use Neural Networks (NNs) to learn to convert colors from one space to another space. We note that from an engineering point of view, this method is probably not the best and most efficient way of converting the colors from one space to another. However, this is immaterial since our focus is not finding the best solution for this color conversion, or even a "correct" solution (a correct solution being what reflects the convention in conversion between the two spaces). Our focus is rather on isolating and studying the effect of different confidence methods in the entire process.

This experiment is inspired by how we learn to map colors from a visual space (what we see) to a linguistic space (what we call the colors). In doing so, there are no one universal "correct" answer as what an English speaker calls *red* is called *rot* in German, *rosso* in Italian, *ghermez* in Persian, and *ahmar* in Arabic. However, each of these communities internally reached a collective agreement on what to call different colors based on an implicit communal self-learning process. Lack of such harmonious agreements leads to a linguistic split in the community, resulting in different dialects and eventually different languages, if the discord increases.

Here, two qualities emerge as positive metrics in such a condition. One is whether the community is harmonious (using the same word for the same concept) or are there many disagreements (different words for the same concepts). Another factor is how fast they can achieve such a harmony, that is, e.g., to come up with a new word for new concepts and the rate of usage spread among the community. In such a context, each individual learns from its environment (community) what words to use for a given concept, and based on their confidence; they adopt new words to conform with the community or insist on specific vocabulary. The latter, which may or may not correspond to overconfidence, may lead to three interesting results: (i) establishment of new vocabulary, (ii) estrangement of individual and small disharmony, (iii) split between the community on how to call a concept and a considerable disharmony.

Inspired by that, in our experiments, the NNs are designed to map from one color space to another color space, which may lead to a different truth and agreement for each community. That is, what may be mapped by a community to red may be mapped to blue by another community, regardless of its "true" class (which may be green). However, the important aspect is that the only difference between communities in an experiment is the confidence method they are using. Therefore, what we see in the community's collective behavior in terms of speed of convergence, and remaining consistent and harmonious after convergence, represent the qualities of that particular confidence method in such a learning context. A useful metric for self-assessment of confidence by each community member should lead to each member's fast and dynamic learning, leading to a harmonious and stable agreement.

Although this experiment is simplified and from an engineering perspective may not be the most efficient, this work and its fundamental concepts are expected to extend to more complex contexts and applications. For example, a community of independent robots that discover their environment by self-learning and need to reach various agreements regarding their environment and courses of action to achieve a common objective can benefit from this concept.

II. Related Works

Authors in [1] define confidence as "An extent to which a procedure may yield the same results on repeated trials." According to their definition if $E_{x_i}^{A_j}$ is the estimation of algorithm *A* for x_i (a sample(s) that belong(s) to class k_l), and $T_{x_i}^{k_l}$ is the ground truth, then the confidence of *A* for that estimation is:

$$c\Big(A(x_i, k_l)\Big) = p\Big(E_{x_i}^{A} == T_{x_i}^{k_l}\Big), \tag{1}$$

Where p is the probability of the estimation being equal to the ground truth. Thus, the authors here have used probability (Section IV-A) as their approach to confidence. The average of confidence c over all the samples gives us the overall confidence of the algorithm, i.e.,

$$C_A = \frac{1}{m}\sum_{k_l=0}^{m} c(A^{k_l}), \tag{2}$$

where m is the total number of classes and

$$c\Big(A^{k_l}\Big) = \frac{1}{n}\sum_{i=0}^{n} c\big(A(x_i, k_l)\big), \tag{3}$$

where n is the total number of samples classified as belonging to each class during the cross-validation.

However, there are different approaches to implement the concept of confidence in machine learning processes [2]–[4] and in CSA systems in general [5], [6]. In [2], authors have applied confidence for their research on Epileptic Seizure Detection, although they did not define it explicitly. They choose a black-box approach using a Support Vector Machine (SVM) to model their confidence. Even though not explicit, we can consider them in-line with the distance method, since fundamentally that is how SVM operated. In the distance method, e.g., used in [6], the system's confidence is the estimated distance of its result with the ground truth.

In [4], authors goal is to be able to factorize the AlexNet to get a reduction in computation load [7], which requires a massive 0.7-1.0G Floating Point Operations (FLOPs) per classification. Their approach is to break the CNN of AlexNet to a set of much smaller Micro CNNs (μCNNs) that are fed by various sub-band inputs sampled from the original input image using Discrete Wavelet. Then, based on the confidence of the μCNNs, which is the probability of correct classification, the number of layers used for the classification is determined. That is, they start with a small number of μCNNs, if the confidence

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test

of the classification is not enough, they add another μCNN at each step.

We have gathered a summary of the works presented above and the effect of using confidence in Table I. Lastly, we note that as discussed in [8], there is a third possibility, which is a combination of both. That is the probability of being within a certain distance from the ground truth. Even though this approach has not been explicitly applied in the literature before, we study it in our work to evaluate its potentials.

TABLE I: Summary of confidence-based ML works

Work	Approach	Comments
[2]	SVM	Improved detection, less power consumption
[3]	Probability	Improved reliability, better/similar detection
[4]	Probability	Computation load reduction, similar detection
[6]	Distance	Improved detection, increased robustness

III. Experiment Setup

We have devised a color conversion experiment to investigate and visualize the effect of confidence when defined based on probability, distance, or both. The purpose of each agent is to convert a random color code from one space to another. As discussed in Section I, the objective is not to find the best or even a "correct" solution for this conversion. The objective is to isolate and evaluate the effect of different confidence methods in a learning context, here a mapping problem. Therefore, the point is to assess how learner agents achieve this (creating a solution for mapping from one space to another) as a community. In the experiments, each learner agent shares with the community what conversion it believes to be correct. Then, based on the confidence of each agent (taken into account using one of the methods for expressing confidence) the community decides about the (virtually) "true" conversion. The community then feeds back all the agents with its decision to train accordingly. We contend that how the community evolves can provide insight into the nature of confidence, at least from a practical engineering perspective, in a communal unsupervised learning context. That is, which one leads to a "better" community? A "better" community, (i) can sooner reach an agreement and (ii) remain in agreement. The speed of convergence is a metric to consider as desired since it shows how nimble agents and the overall community is in learning. Remaining in agreement, on the other hand, shows how well agents have learned what the community has learned.

A. Agent Model Definition

In our experiments, we have decided to use untrained NNs (initialized with random numbers) as the model of agents, who need to come to an agreement with their community regarding a classification problem. We have decided to use Keras API [9] for implementing our agents because of its ease of use. Our models consist of two hidden layers, each with eight fully-connected nodes. The input layer has three nodes, and the output layer's nodes vary following the mapping problem (Figure 1). In this experiment, we have datasets of 10000 randomly generated three-dimensional values as our input, which we newly generate for every single epoch to fabricate a random environment for the network. They would go in as individual dimensions (numbers) to the three nodes of the input layer. The outputs are multi-dimensional values coming out of the network as individual numbers (e.g., three-dimensional for a network with three nodes at its output layer). As also shown on Figure 1, the number of output nodes in this work is a member of the $\{3, 4, 6, 9\}$.

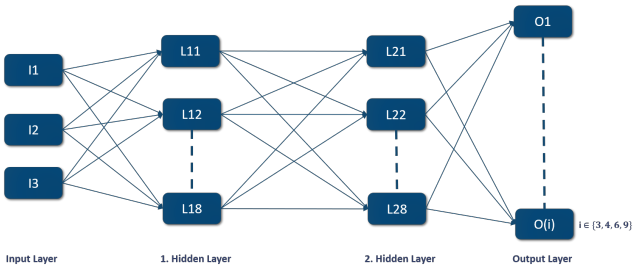


Fig. 1: Implemented Neural Network Architecture.

We are using Mean Squared Error (MSE) as our loss function that is commonly used for regression to values between 0 and 1, in combination with the Sigmoid function as the activation for the last layer (output layer) as suggested in [9]. The activation for hidden layers is the Rectified Linear Unit (ReLU) function. Using sigmoid as our last layer activation would bring us positive zero to one values that we take as predicted normalized output values. For the compilation part, we studied different optimizers and decided for the Adam optimizer [10] with a learning rate of 0.001.

B. Task Definition

Our agents are supposed to convert colors from one space to another space. Normally for this procedure, we need a dataset as our input and target values (or labels) that should be the correctly converted values of those inputs. The agents then use the target value for the weight update in the backpropagation. However, as discussed before, a correct conversion is not our objective. Hence, in our socially-inspired experiment, we assume that the community does not know the true conversion and needs to come into agreement about how to classify the inputs to certain outputs. Hence, instead of feeding the models with the ground truth, we use target values generated by the community itself. We take the agents' predictions in the corresponding community after every single epoch and use them to construct the target values used for training the agents on converting the color codes. How these target values are constructed for training is our window into the nature of confidence investigation, which we discuss in Section IV.

C. Communities

We conduct eighteen experiments, each consisting of two or more of the agents described in Section III-A. The experiments include communities with 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 agents. The experiments, or equivalently communities, are named based on the number of members they have. That is, Community 2 has two members (two NN agents).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test

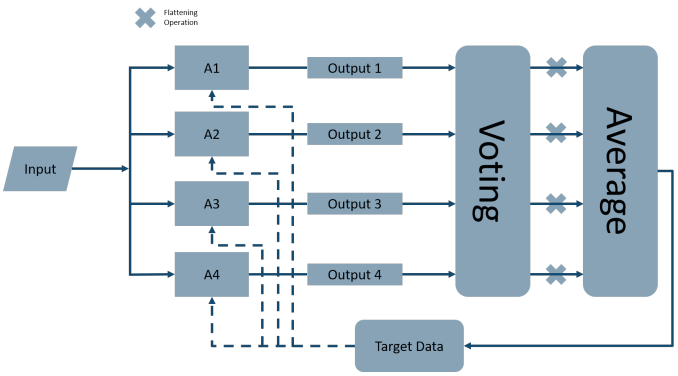


Fig. 2: Block diagram of the Probability method for Community 4.

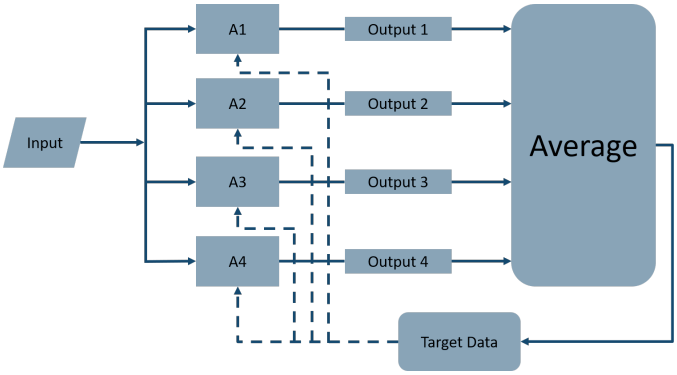


Fig. 3: Block diagram of Distance method for Community 4.

IV. Confidence Methods

After we designed our communities, as mentioned in Section III, we now present our specific confidence-based methods of constructing target values (labels). This way, we can evaluate how they affect the performance of our communities.

A. Probability

In this experiment, each agent in the community casts a probabilistic vote of confidence in classifying each input into one of the (3, 4, 6 or 9) output bins. That is, each agent casts its vote based on the probabilistic measure of how confident it is about its classification. The community then decides what the class of that input is, based on all its agents' votes. This decision is then taken as the label of the input and backpropagated.

To this end, first we search for the dominant value in agents' prediction for each class and take their dominant value as their votes. Then, we check for the most voted class. The community's target value, T_C, would then be calculated as the average of the voted values from all agents. With j referring to the output bin index, and k being the chosen bin (B^k), and *i* referring to the agent number and *n* being the total number of agents we will have

$$T_C^j = \begin{cases} \frac{1}{n}\sum_{i=1}^{n} B_i^j & : \ j = k, \\ 0 & : \ j \neq k \end{cases} \tag{4}$$

The block diagram of the probability method can be found in Figure 2. We note that in this method, only one of the classes (the one that the community voted for) will have a non-zero value, and the other are set to zero. For example, if Agent 1 has an output of (0.8, 0.9, 0.3) and Agent 2 has an output of (0.2, 0.7, 0.3), then both are voting for the second class (T_C^2) and T_C=(0, 0.8, 0). However, if Agent 2 has an output of (0.7, 0.2, 0.3), it is voting for the first class (T_C^1), and there is a tie between class one and class two. In this case, the average of all colors leads to (0.75, 0.55, 0.3), and since the first class (T_C^1) has a higher average, the final vote is the first class and other classes are set to zero, that is T_C=(0.75, 0, 0).

B. Distance

In this method, an agent's confidence is based on how far its prediction is from the value it is supposed to have. Each agent then learns and adjusts its predictions based on the distance from the communally decided target value. We gather learner agents' predictions after each run, and we calculate the target value (label) for the input in a communal fashion. Each agent in a community has a decision, $D = (B^1, B^2, ..., B^j)$, on the provided data. Hence, the label or target value of the community in this method is decided by

$$T_C = \frac{1}{n}\sum_{i=0}^{n} D_i. \tag{5}$$

Figure 3 shows the block diagram of this method. In summary, the target value (label) is the average of predicted value of all classes of all agents. In the two examples of Section IV-A, the T_C=(0.5, 0.8, 0.3) for the first example and T_C=(0.75, 0.55, 0.3) for the second example.

C. Flattened Distance

The idea behind this method is to bring the distance method closer to the probability method and, in a way, construct a combination of distance and probability methods. To this end, we first flatten our data in hand, which means we pick the dominant value between the output bins (colors) and set the others to zero. It is saying that when the B^k value of a color code is the biggest of all, it means that it has the closest distance to the pure B^k than to other bins (colors)[1]. Thus, each agent's contribution to the community is by expressing their distance from the pure color of their dominant color. Then, we categorize the colors to its nearest defined label. After that, the flattened data from each agent is averaged and fed to the network for learning.

We consider agent(s)' outputs as our output elements, *O*. In a generic case where a community has *n* output elements and the target value of the network is T_C, the distance algorithm is described by Algorithm 1. Figure 4 shows the block diagram of this method. In this method, the target value (label) is the average of predicted value of only one class (the dominant color) of each agent. For instance, in the two examples of

[1]Similar to the concept of SVM

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test

Algorithm 1: Flattened Distance Algorithm

Input: O_i
Output: T_C

for $i = 1$ *to* n **do**
  if B^k *has the highest value* **then**
    $O_i = (0, 0, ..., B_i^k, 0, 0)$
  end
  $T_C = \frac{1}{n} \sum_{i=1}^{n} O_i$
end

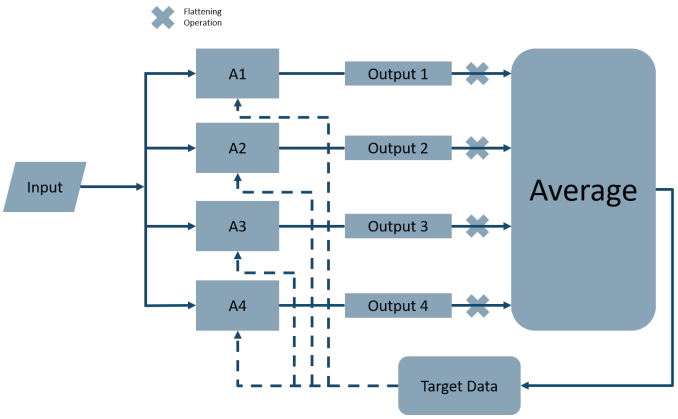


Fig. 4: Block diagram of Flattened Distance method for Community 4.

Section IV-A, the T_C=(0, 0.8, 0) for the first and T_C=(0.35, 0.45, 0) for the second example.

V. Results

To evaluate the effect of the described confidence methods, we ran our experiments fifty times, each run including 60 epochs. We then averaged the performance of each community at each epoch during the fifty runs.We repeated these experiments for different number of output bins, namely, 3, 4, 6, and 9. Figure 5 shows the average performance result of four of the communities with four output bins using each of the three confidence methods. For a better demonstration of results, we chose the four communities that would best represent the trend we observed in the experiments. In particular, regarding the changes related to the size of the communities.

As we see in Figure 5, the probability method has the best "accuracy" in all experiments. Meaning that in communities where members use probability as their confidence while contributing to their communal learning (on deciding how to classify certain inputs) remain more harmonious and will continue to have similar opinions. They also require, a shorter convergence time, which means such communities need fewer interactions to reach their best harmonious state (which is -in most cases- more harmonious than other communities).

The distance method, however, seems to degrade generally in performance in larger communities. In particular, the convergence time considerably increases in this method. It is also interesting to note that the flattened distance method, even though it starts slow and with the largest disharmony, quickly catches up and surpasses the distance method in the smaller communities with a smaller number of output bins. As the number of output bins or the number of community members increases, the performance of this method considerably degrades.

VI. Discussion and Conclusion

Our experiments can also be thought of as a voting ensemble. Voting ensemble is a consensus-based mechanism that combines the decisions of multiple independent classifiers in an attempt to increase the classification accuracy [11]. As different voting methods, we can mention the regression voting ensemble (average of the models' predictions) and the classification voting ensemble (what the majority of models are voting for). Our distance method is similar to the regression voting ensemble and our probability method is a combination of both them. However, our goal is not really increasing the accuracy of classification, since in our experiments we do not consider any ground truth to be able to define any classification accuracy in the usual way.

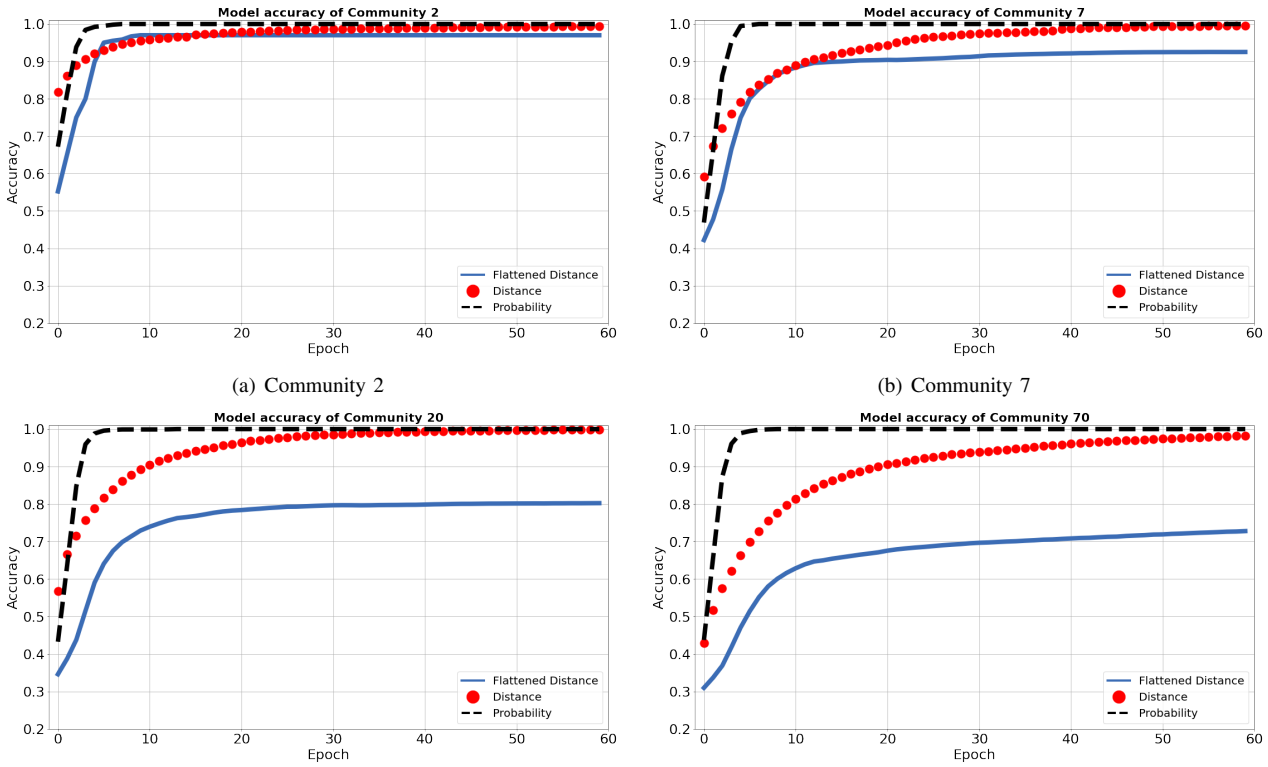
In our experiments, NN agents contribute to a communal decision and learn from that communal decision to correct their own behavior (to conform to the community). Both of these two actions are based on the agent's confidence, for which we employed three different mechanisms. Here, the speed of reaching a communal harmony and the level of agreement, to which this harmony converges, reflect the quality of the mechanism used by agents for assessing their confidence. A good mechanism of confidence allows an agent to have a better, more realistic view of its own capabilities than its environment and thus learn as fast as possible (speed of convergence). Moreover, it allows an agent to recognize its mistakes, admit and correct them, which leads to a more harmonious community at the end. We note that even though differences of opinion can be helpful, in specific contexts and communities and generally speaking for machine learning purposes is not considered positive.

Based on these considerations, our experiments show that probability is a better mechanism for confidence. We take this as an indication of what may the nature of computational confidence be. More importantly, which mechanism may be more helpful in engineering contexts where confidence is used, particularly in multi-agent (self-)learning contexts. Nevertheless, we do not forget that this is only one evaluation experiment among many possible evaluation methods. However, we believe this is a good starting point for the community to understand the nature of confidence better and explore various evaluation methods that can shed more light on the nature of the confidence and what mechanisms might be more helpful for systems using confidence. Importance of which are discussed in many works, including [1], [6], [8], [12], [13].

References

[1] N. TaheriNejad et al. Comprehensive observation and its role in self-awareness; an emotion recognition system example. In *FedCSIS*, 2016.
[2] Farnaz andothers Forooghifar. Self-aware wearable systems in epileptic seizure detection. In *21st Euromicro Conference on DSD*, pages 426–432. IEEE, 2018.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MDAT.2021.3078341, IEEE Design and Test



(a) Community 2

(b) Community 7

(c) Community 20

(d) Community 70

Fig. 5: A demonstration of communities performance while applying confidence with four output bins in the form of the three presented methods.

[3] H. A. Kholerdi et al. Enhancement of classification of small data sets using self-awareness; an iris flower case-study. In *IEEE ISCAS*, pages 1–5, 2018.

[4] Katayoun Neshatpour et al. Icnn: An iterative implementation of convolutional neural networks to enable energy and computational complexity aware dynamic approximation. In *DATE*, pages 551–556. IEEE, 2018.

[5] M. Götzinger et al. Confidence-enhanced early warning score based on fuzzy logic. *ACM/Springer Mobile Networks and Applications*, pages 1–18, 2019.

[6] M. Götzinger et al. Model-free monitoring with confidence. *International Journal of Computer Integrated Manufacturing*, 32(4-5):466–481, 2019.

[7] Alex Krizhevsky et al. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] N. TaheriNejad and A. Jantsch. Improved machine learning using confidence. In *IEEE 32nd CCECE*, pages 1–5, 2019.

[9] Francois Chollet. *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG, 2018.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Omar A Alzubi, Jafar A Alzubi, Sara Tedmori, Hasan Rashaideh, and Omar Almomani. Consensus-based combining method for classifier ensembles. *Int. Arab J. Inf. Technol.*, 15(1):76–86, 2018.

[12] L. Esterle and J. N. A. Brown. The competence awareness window: Knowing what i can and cannot do. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 62–63, 2020.

[13] J. N. A. Brown and L. Esterle. I'm already optimal: the dunning-kruger effect, sociogenesis, and self-integration. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 82–84, 2020.

Amid Mozelli (amid.mozelli@tuwien.ac.at) is currently an MSc student of electrical engineering and information technology at TU Wien, Vienna, Austria. After he finished his bachelor studies at the Institute of Computer Tbechnology, he has been working as a researcher in the field of machine learning in the Christian Doppler Laboratory for Embedded Machine Learning. His research is focused on optimizing machine learning models on GPU devices, especially NVIDIA platforms.

Nima TaheriNejad (nima.taherinejad@tuwien.ac.at) received his Ph.D. degree in electrical and computer engineering from The University of British Columbia (UBC), Vancouver, Canada, in 2015. He is currently an assistant professor at the TU Wien (formerly known as Vienna University of Technology as well), Vienna, Austria, where his areas of research interest include self-awareness in resource-constrained cyber-physical (embedded) systems, in-memory computing, memristor-based circuit and systems, and health-care. He is a member of the ACM and IEEE Circuits and Systems Society as well as IEEE Engineering in Medicine and Biology Society.

Axel Jantsch (axel.jantsch@tuwien.ac.at) received the Dipl. Ing. and Dr.Tech. degrees from the TU Wien, Vienna, Austria, in 1988 and 1992, respectively. He was full Professor (2000-2014) at the Royal Institute of Technology (KTH), Stockholm, Sweden and since 2015 he has been Full Professor of Systems on Chip at the TU Wien, Vienna, Austria. He has published over 300 papers in international conferences and journals, and several books. His current research interests are on embedded machine learning and self-aware cyber-physical systems.