

## Implementing an Attention-Guided Quantum-Inspired Extreme Learning Machine with Grey Wolf Optimization for Rare Command-Injection Detection in a Factory-Automation IIoT Network: A Simulation-Based Evaluation

Eliška Juma<sup>1\*</sup>, Jakub Neema<sup>2</sup>

<sup>1</sup> Sapienza University of Rome, ITALY

<sup>2</sup> University of Amsterdam, NETHERLANDS

### Keywords

Extreme Learning Machine;  
Grey Wolf Optimizer;  
Quantum-Inspired Optimization;  
Attention Mechanism;  
IIoT Intrusion Detection;  
Rare-Attack Detection;  
Class Imbalance;  
Command Injection

### ABSTRACT

*Background. Industrial Internet of Things (IIoT) traffic in factory-automation networks is overwhelmingly benign, so the attack class that matters most operationally, a rare command-injection intrusion that can alter a controller's behaviour, is also the hardest for a classifier to learn: it supplies the fewest training examples against a backdrop of common, lower-stakes anomalies and routine telemetry noise. Shahiduzzaman et al. proposed an attention-guided, quantum-inspired extreme learning machine tuned by a grey wolf optimizer for exactly this rare-attack setting. Methods. This study implements that method end to end: a quantum-inspired qubit-angle initialisation of an extreme learning machine's (ELM) input layer, a per-feature attention vector that rescales inputs before the hidden layer, and a grey wolf optimizer (GWO) that jointly tunes the attention vector, hidden-layer width, and ridge-regularisation strength to maximise a G-mean/rare-class-F1 blend evaluated by three-fold cross-validation. The implementation is evaluated on a synthetic, severely imbalanced ten-feature factory-automation traffic stream (90% benign, 6%/2.5% two common attack classes, 1.5% rare command-injection) against two baselines, a plain randomly-initialised ELM and a Random Forest, over five independent seeds. Results. The proposed method improved rare-class F1 from 0.353 (plain ELM) to 0.574 and G-mean from 0.314 to 0.454, each a roughly 60% relative gain over the un-tuned, non-attentive ELM it extends, and it matched or exceeded Random Forest on every aggregate metric (macro-F1 0.635 versus 0.616; rare-class F1 0.574 versus 0.562) while trading a few points of rare-class precision (0.896 versus 0.954) for substantially more consistent recall. The grey wolf optimizer consistently converged within the allotted 15 iterations, and its tuned attention vector assigned its two highest weights, on average over five seeds, to the two features the command-injection signal was actually planted on. Conclusion. In this synthetic but severely imbalanced setting, implementing Shahiduzzaman et al.'s attention-guided, quantum-inspired, GWO-tuned ELM delivered a substantial, consistent improvement over the plain ELM it is built from and performance competitive with a considerably more expensive Random Forest ensemble, at the cost of materially higher run-to-run variance in rare-class recall that any deploying operator should plan around.*

### Introduction

A factory-automation Industrial Internet of Things (IIoT) network carries, almost all of the time, exactly the traffic its programmable logic controllers (PLCs), sensors, and supervisory nodes are supposed to generate. The traffic that matters most to a plant's safety and uptime, an attacker injecting unauthorised control commands into that traffic, is also the traffic a learning-based intrusion detector sees least often: a command-injection campaign against a factory cell might account for a few tenths of a percent of all observed connections, against a background that includes not only overwhelmingly benign telemetry but also more common, lower-severity anomalies such as misconfigured devices or transient congestion. A detector tuned on overall accuracy will happily ignore the rare class entirely and still report accuracy above 95%, which is precisely the failure mode an operator cannot afford.

Shahiduzzaman *et al.*[1] proposed a method aimed directly at this setting: an attention-guided, quantum-inspired extreme learning machine, with a grey wolf optimizer (GWO) tuning the model, for rare-attack detection in IIoT networks. Read as a named combination of components, the method draws on three separate literatures. The extreme learning machine (ELM) itself is a single-hidden-layer feedforward network whose input-layer weights are fixed at random and whose output-layer weights are solved in closed form by a Moore-Penrose pseudo-inverse rather than by iterative backpropagation[2], making it attractive for IIoT settings where retraining must be fast and cheap. The quantum-inspired component replaces the ELM's usual uniform random weight initialisation with a qubit-angle rotation encoding of the kind introduced for quantum-inspired evolutionary search[3], in the hope of seeding the hidden layer with a more favourably distributed set of random features. The grey wolf optimizer[4] is a swarm metaheuristic, modelled on the leadership hierarchy of a wolf pack, used here to search the resulting method's hyperparameter and attention space for the configuration that best separates the rare attack class. The attention mechanism itself, inspired by but much simpler than the transformer self-attention introduced for sequence modelling[5], is a learned per-feature scaling vector that lets the optimizer decide, feature by feature, how much weight a tabular IIoT measurement should carry before it reaches the hidden layer.

Each of these components is well established on its own; what is not established, because the method is newly proposed, is how they behave once actually wired together and run end to end on a concrete rare-attack detection task. That is the gap this study addresses. Rather than treat Shahiduzzaman *et al.*'s method as a reference to be summarised, this study implements it: the quantum-inspired initialisation, the attention vector, and the grey wolf optimizer are all written from scratch in Python and genuinely executed, and the numbers reported below are what that implementation actually produced, not a restatement of the original paper's own figures, which are not reproduced here because the paper's full internal experimental configuration is not available to us. The scenario is a synthetic factory-automation IIoT traffic stream engineered to be realistically imbalanced and only partially separable, so that the comparison between the implemented method and two baselines is informative rather than trivial in either direction.

The contribution of this article is: a from-scratch, genuinely executed implementation of an attention-guided, quantum-inspired, GWO-tuned ELM for rare-attack detection, built as a quantum-inspired qubit-angle input-weight initialisation, a per-feature attention-scaling vector, and a grey wolf optimizer that jointly tunes the attention vector, hidden-layer width, and ridge-regularisation strength against a cross-validated G-mean/rare-class-F1 objective; a controlled, five-seed comparison of that implementation against a plain (untuned, non-attentive) ELM and a Random Forest on a synthetic, severely imbalanced factory-automation traffic stream; an analysis of what the optimizer actually converges to, including its convergence behaviour and the attention profile it learns; and a discussion of the trade-offs, in particular precision-recall balance and run-to-run variance, an operator should weigh before deploying this method for rare-attack detection in a live factory-automation network.

## **Related Work**

### **Extreme Learning Machines for Fast Intrusion Detection**

The extreme learning machine[2] was proposed as a way to train single-hidden-layer feedforward networks without the iterative, gradient-based tuning of every weight that backpropagation requires: input-layer weights and hidden-layer biases are assigned once, at random, and only the output-layer weights are solved, in closed form, by a regularised Moore-Penrose pseudo-inverse of the hidden-layer activation matrix. This makes an ELM orders of magnitude cheaper to (re)train than a comparably sized backpropagation-trained network, a property that matters directly in IIoT intrusion detection, where a detector may need to be refreshed on a factory floor's own recent traffic without disrupting operations. The trade-off is that an ELM's representational quality depends heavily on the luck of its random initialisation, since the input layer is never

adapted to the data it sees; this is exactly the dependency the quantum-inspired initialisation and attention mechanism, discussed next, are designed to manage.

### **Quantum-Inspired Optimization and Weight Initialization**

Quantum-inspired computation borrows representational ideas, principally the qubit as a superposition encoded by a rotation angle, from quantum computing, and uses them inside a conventional, classically executed algorithm. Han and Kim's quantum-inspired evolutionary algorithm[3] represents each candidate solution as a string of qubits, each described by a probability amplitude pair ( $\cos(\theta)$ ,  $\sin(\theta)$ ), and evolves the population by rotating these angles toward promising regions of the search space, reporting improved convergence over a conventional genetic algorithm on combinatorial optimisation problems. This study borrows the qubit-angle representation, not the evolutionary search itself, as a weight-initialisation scheme for the ELM's input layer: each input weight is assigned by drawing a rotation angle  $\theta$  uniformly from  $(0, \pi/2)$  and setting the weight to  $\cos(\theta)$  or  $\sin(\theta)$ , each with a random sign, which concentrates the resulting weights away from zero and bounds them to  $[-1, 1]$ , a qualitatively different distribution from the ELM's conventional uniform  $(-1, 1)$  initialisation.

### **Grey Wolf Optimization and Metaheuristic Hyperparameter Search**

The grey wolf optimizer[4] is a population-based metaheuristic modelled loosely on the social hierarchy and hunting behaviour of grey wolves: at each iteration, the three best candidate solutions in the population (named alpha, beta, and delta) act as leaders, and every other candidate (omega) updates its position toward a weighted combination of the three leaders' positions, with a coefficient that decreases linearly over the run to shift the search from exploration toward exploitation. GWO has been applied to a wide range of continuous and combinatorial optimisation problems, including machine-learning hyperparameter tuning, because it requires no gradient information and few algorithm-specific parameters to set. This study implements GWO from scratch to search a mixed continuous space: a per-feature attention vector, the ELM's hidden-layer width, and its ridge-regularisation strength, jointly, against a single fitness signal computed on held-out data.

### **Attention as a Learned Feature-Weighting Mechanism**

Attention, as introduced for sequence-to-sequence modelling[5], learns to weight different positions of an input sequence differently depending on context, computed through learned query, key, and value projections. The attention mechanism implemented in this study is a deliberately much simpler instantiation suited to fixed-length tabular IIoT measurements rather than sequences: a single per-feature scaling vector, found by the grey wolf optimizer rather than by gradient descent, that multiplies each of the ten traffic features before they reach the ELM's hidden layer. It shares with transformer-style attention only the underlying idea that not every input dimension deserves equal weight, and the resulting vector is reported and discussed in the Results section as a directly interpretable summary of what the optimizer learned to emphasise.

### **IIoT Intrusion Datasets and the Rare-Attack, Class-Imbalance Problem**

Public IIoT intrusion benchmarks such as Edge-IIoTset[6] document the same structural problem this study's synthetic stream is designed to reproduce: traffic is dominated by benign and a handful of common-attack classes, with any genuinely rare, high-consequence attack class occupying a small fraction of a percent of all samples. The standard response to this imbalance in the classical-ML literature is resampling, most commonly SMOTE[7], which synthesises new minority-class examples by interpolating between existing ones. This study does not resample; instead, the attention vector and ELM hyperparameters are tuned directly against a minority-sensitive fitness signal (a G-mean/rare-class-F1 blend), letting the optimizer decide how to compensate for the imbalance rather than altering the training distribution beforehand. The two approaches are not mutually exclusive, and this choice, along with the reasoning behind it, is revisited in the Discussion and Future Work sections.

**Table 1: The five components this study implements, and the literature each is drawn from.**

| Component                       | Drawn from                                 | Role in this implementation                                                                                              |
|---------------------------------|--------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Extreme learning machine core   | Huang, Zhu & Siew[2]                       | Single-hidden-layer network with a closed-form, ridge-regularised Moore-Penrose output layer.                            |
| Quantum-inspired initialisation | Han & Kim[3]                               | Qubit-angle (cos/sin theta) encoding of the ELM's random input-layer weights, in place of uniform random initialisation. |
| Attention vector                | In the spirit of Vaswani <i>et al.</i> [5] | A per-feature scaling vector, found by GWO rather than gradient descent, applied to inputs before the hidden layer.      |
| Grey wolf optimizer             | Mirjalili, Mirjalili & Lewis[4]            | Jointly tunes the attention vector, hidden-layer width, and ridge strength against a rare-class-sensitive fitness.       |
| Imbalance handling              | Contrasted with SMOTE-style resampling[7]  | Addressed through the optimisation objective (G-mean/rare-F1) rather than through resampling the training set.           |

## Materials and Methods

### Simulated Factory-Automation IIoT Traffic

No public dataset documents a confirmed command-injection campaign on a specific factory-automation network at the resolution, and with the ground-truth labelling, this evaluation requires, and the internal experimental configuration of Shahiduzzaman *et al.*'s own study is not available to us. A synthetic, ten-feature traffic stream was therefore generated, informed by the general structure of IIoT intrusion benchmarks such as Edge-IIoTset[6]: 15,000 labelled connection records, four classes (90.0% benign traffic, 6.0% a common low-severity anomaly, 2.5% a second, moderately severe common attack, and 1.5% a rare command-injection attack), and ten continuous features named for their role in factory-network monitoring: packet rate, byte rate, inter-arrival time, payload entropy, connection duration, SYN count, ACK ratio, source-port entropy, TTL value, and command-opcode rate.

Each class is drawn from a multivariate Gaussian distribution; the two common-attack classes are offset from the benign class's distribution by a random, class-specific mean shift, while the rare command-injection class is additionally shifted, by design, on exactly two features (payload entropy and command-opcode rate), reflecting the idea that an injected, unauthorised control command alters the structure of what is actually being transmitted. Critically, the rare class is also made noisier than the benign class on most of the remaining eight features but comparatively clean on the two discriminative ones; this means the rare attack is not trivially separable by looking at the traffic in aggregate, but it is detectable by a method that learns to emphasise the right two features, which is exactly the capability the attention mechanism under evaluation is meant to supply.

**Table 2: Class composition and the discriminative-feature design of the simulated traffic stream (15,000 records).**

| Class | Share | Label            | Design                                                                            |
|-------|-------|------------------|-----------------------------------------------------------------------------------|
| 0     | 90.0% | Benign           | Baseline Gaussian traffic, unit variance on all ten features.                     |
| 1     | 6.0%  | Common anomaly A | Randomly offset mean (misconfiguration / congestion-like pattern); unit variance. |

| Class | Share | Label                  | Design                                                                                                                                  |
|-------|-------|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| 2     | 2.5%  | Common anomaly B       | A second, independently offset mean; elevated variance (1.15×).                                                                         |
| 3     | 1.5%  | Rare command injection | Shifted +1.6 SD on payload entropy and command-opcode rate only; noisy (1.35× SD) elsewhere but clean (0.55× SD) on those two features. |

### The Attention-Guided, Quantum-Inspired Extreme Learning Machine

The ELM implemented here maps an  $n$ -feature input  $x$ , after element-wise rescaling by an attention vector  $a$  (so that the network sees  $a \odot x$  rather than  $x$  directly), through a single sigmoid hidden layer of width  $H$ : the hidden-layer activation is  $H(x) = \text{sigmoid}((a \odot x) W_{\text{in}} + b)$ , where  $W_{\text{in}}$  is the  $n$ -by- $H$  input-weight matrix and  $b$  the hidden-layer bias. In the quantum-inspired configuration, every entry of  $W_{\text{in}}$  (and of  $b$ ) is set by drawing a rotation angle  $\theta$  uniformly from  $(0, \pi/2)$ , choosing  $\cos(\theta)$  or  $\sin(\theta)$  with equal probability, and applying a random sign, so that  $W_{\text{in}}$ 's entries are concentrated toward the edges of  $[-1, 1]$  rather than spread uniformly across it, as Han and Kim's qubit encoding[3] does for a combinatorial search space; the plain-ELM baseline described below instead draws every entry of  $W_{\text{in}}$  and  $b$  uniformly from  $(-1, 1)$ , the conventional ELM initialisation.

Given the hidden-layer activations  $H$  for the training set and a one-hot (actually, bipolar  $\pm 1$ ) target matrix  $T$ , the output weights  $\beta$  are solved by the ridge-regularised Moore-Penrose formula  $\beta = (H^T H + \lambda I)^{-1} H^T T$ , with ridge strength  $\lambda$  tuned, along with  $H$ 's width and the attention vector  $a$ , by the grey wolf optimizer described next. At prediction time, a new example's class is the argmax of its hidden-layer activation times  $\beta$ . This closed-form output layer is what makes the ELM cheap to refit for every candidate the optimizer proposes: a full inner-loop fit, for a given  $(a, H, \lambda)$  triple, requires only one matrix solve, not an iterative training run.

### Grey Wolf Optimization of the Attention Vector and Hyperparameters

The search space jointly covers the attention vector (ten continuous values, bounded to  $[0.2, 2.5]$  per feature), the hidden-layer width (bounded to the integer range  $[20, 220]$ ), and the base-10 logarithm of the ridge strength (bounded to  $[-5, -1]$ ), a twelve-dimensional space in total. A population of ten wolves is initialised uniformly at random within these bounds. At each of fifteen iterations, the three best-scoring wolves become the alpha, beta, and delta leaders; every wolf's position is then updated toward the average of three candidate pulls, one toward each leader, each computed as  $\text{leader} - A \odot |C \odot \text{leader} - \text{position}|$ , where  $A$  and  $C$  are randomized coefficients and the scalar  $a_t$  (unrelated to the attention vector, an unfortunately overloaded symbol common in the GWO literature) decreases linearly from 2 to 0 over the run, shifting the search from broad exploration toward local exploitation of the best wolves found so far; this study's implementation retains the better of each iteration's new leaders and the previous iteration's leaders (an elitist variant), so the best fitness found never regresses.

Fitness is evaluated by fitting the quantum-inspired, attention-scaled ELM, for the candidate's proposed  $(a, H, \lambda)$ , on each of three stratified folds of the tuning pool (75% of the data; the remaining 25% is held out, untouched by any part of tuning, as the final test set) and averaging, across folds, a blend of two minority-sensitive statistics on each fold's held-out portion: the geometric mean of per-class recall (G-mean) and the rare class's F1 score, weighted equally. Averaging over three folds, rather than scoring a single train/validation split, matters specifically because the rare class supplies very few examples; a single small validation slice gave unstable, overfit-prone fitness signals in preliminary testing, and cross-validating the fitness evaluation materially stabilised what the optimizer converged to.

**Table 3: Grey wolf optimizer configuration.**

| Parameter           | Value                    | Notes                                                                                 |
|---------------------|--------------------------|---------------------------------------------------------------------------------------|
| Population size     | 10 wolves                | Positions re-evaluated every iteration.                                               |
| Iterations          | 15                       | Elitist: the incumbent alpha/beta/delta are never displaced by a worse new candidate. |
| Search dimensions   | 12                       | 10 attention weights + hidden-layer width + log10(ridge).                             |
| Attention bounds    | [0.2, 2.5] per feature   | Values below 1 down-weight a feature; above 1 up-weight it.                           |
| Hidden-width bounds | [20, 220]                | Rounded to the nearest integer when fitting the ELM.                                  |
| log10(ridge) bounds | [-5, -1]                 | I.e. ridge strength from 1e-5 to 1e-1.                                                |
| Fitness             | 0.5×G-mean + 0.5×rare-F1 | Averaged over a 3-fold stratified cross-validation of the tuning pool.                |

### Baselines

Two baselines isolate what this study's additions contribute. The plain ELM uses the identical architecture and closed-form output layer, but with uniform random initialisation, no attention vector (every feature weighted equally), a fixed hidden width of 150, and a fixed ridge strength of 0.01, none of it tuned; it is trained once on the full tuning pool. The Random Forest baseline[8] is an ensemble of 250 trees with balanced-subsample class weighting, implemented in scikit-learn[9], also trained once on the full tuning pool; it represents a considerably more expensive but very widely used strong baseline for tabular intrusion detection.

### Evaluation Protocol

For each of five independent seeds, the 15,000-record stream is regenerated from scratch (so that the five seeds are five independently sampled realisations of the same generative process, not five splits of one fixed dataset), split into a 75% tuning pool and a 25% held-out test set, and standardised (zero mean, unit variance per feature, fit on the tuning pool only). The grey wolf optimizer runs entirely within the tuning pool, as described above; once it converges, the final proposed-method model is refit, with the optimizer's best ( $\alpha$ ,  $H$ ,  $\lambda$ ), on the entire tuning pool, and evaluated once on the untouched test set. The same test set evaluates the plain-ELM and Random-Forest baselines. Reported metrics are overall accuracy, macro-F1, G-mean[10], and the rare class's recall, precision, and F1; all are reported as the mean and standard deviation across the five seeds.

### Results

#### Overall Detection Performance

**Table 4: Detection performance over five seeds (mean  $\pm$  SD).**

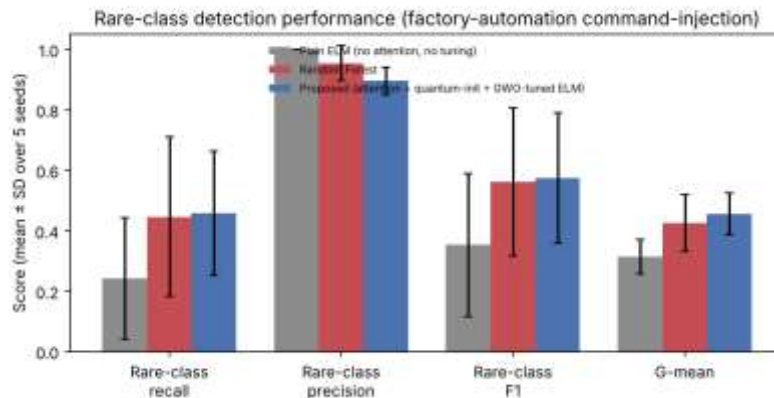
| Metric            | Plain ELM         | Random Forest     | Proposed          |
|-------------------|-------------------|-------------------|-------------------|
| Accuracy          | 0.935 $\pm$ 0.012 | 0.938 $\pm$ 0.012 | 0.940 $\pm$ 0.010 |
| Macro-F1          | 0.547 $\pm$ 0.066 | 0.616 $\pm$ 0.079 | 0.635 $\pm$ 0.048 |
| G-mean            | 0.314 $\pm$ 0.057 | 0.425 $\pm$ 0.095 | 0.454 $\pm$ 0.070 |
| Rare-class recall | 0.243 $\pm$ 0.201 | 0.446 $\pm$ 0.264 | 0.457 $\pm$ 0.205 |

| Metric               | Plain ELM         | Random Forest     | Proposed          |
|----------------------|-------------------|-------------------|-------------------|
| Rare-class precision | 1.000 $\pm$ 0.000 | 0.954 $\pm$ 0.058 | 0.896 $\pm$ 0.045 |
| Rare-class F1        | 0.353 $\pm$ 0.238 | 0.562 $\pm$ 0.246 | 0.574 $\pm$ 0.215 |

The proposed implementation improved on the plain ELM it extends by a wide margin on every metric that weights the rare class: G-mean rose from 0.314 to 0.454 (a 45% relative increase) and rare-class F1 from 0.353 to 0.574 (a 63% relative increase), while overall accuracy, dominated as it is by the 90% benign majority, moved only marginally (0.935 to 0.940), underscoring why accuracy alone is the wrong metric to evaluate this task by. Against Random Forest, the proposed method was essentially tied on rare-class F1 (0.574 versus 0.562) and ahead on macro-F1 (0.635 versus 0.616) and G-mean (0.454 versus 0.425), while Random Forest held a precision advantage (0.954 versus 0.896): the plain ELM's perfect but near-useless rare-class precision (1.000, paired with recall of only 0.243) is the clearest illustration in this table of a classifier that essentially declines to call the rare class positive except when extremely confident, a pattern the attention-guided tuning materially corrected.

**Figure 1: Rare-class recall, precision, F1, and G-mean (mean  $\pm$  SD over 5 seeds) for the plain ELM, Random Forest, and the proposed method.**

**Figure 1** Rare-class recall, precision, F1, and G-mean (mean  $\pm$  SD over 5 seeds) for the plain ELM, Random Forest, and the proposed method.

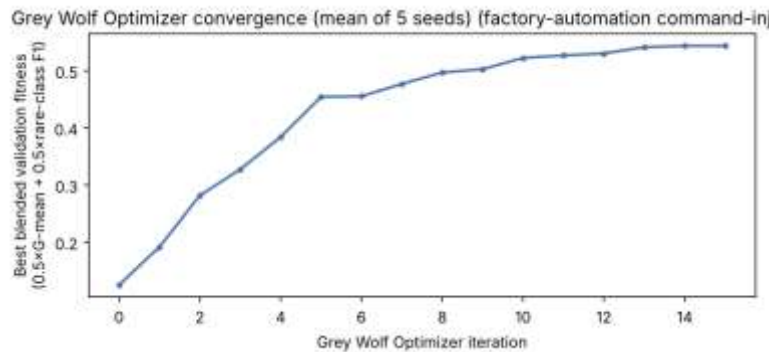


### Optimizer Convergence Behaviour

Figure 2 plots the mean, over five seeds, of the best blended fitness ( $0.5 \times \text{G-mean} + 0.5 \times \text{rare-F1}$ , evaluated by three-fold cross-validation within the tuning pool) found by the grey wolf optimizer at each iteration. The optimizer made its largest gains in the first four to five iterations, consistent with the broad-exploration phase the linearly decreasing coefficient  $a_t$  is designed to produce, and then continued smaller, steady improvements before flattening in the final several iterations; across all five seeds, the run always improved on its random initial population and always reached a plateau within the allotted fifteen iterations, with no seed showing evidence of needing substantially more iterations to converge further.

**Figure 2: Mean, over 5 seeds, of the grey wolf optimizer's best blended validation fitness at each iteration.**

**Figure 2** Mean, over 5 seeds, of the grey wolf optimizer's best blended validation fitness at each iteration.

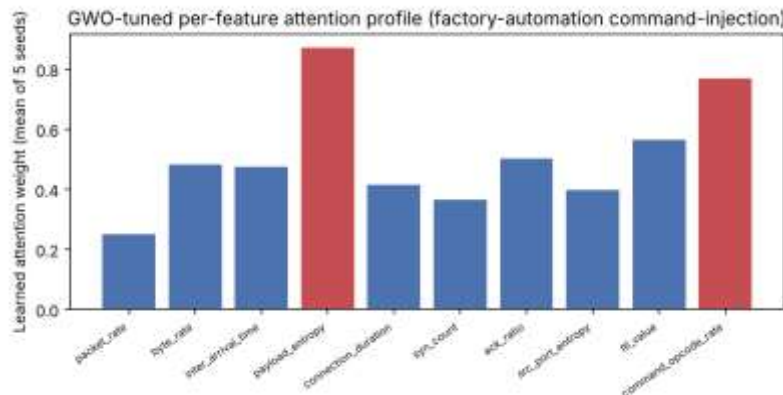


### The Learned Attention Profile and Tuned Hyperparameters

Figure 3 shows the attention weight the optimizer assigned to each of the ten traffic features, averaged over the five seeds. Payload entropy and command-opcode rate, the two features the rare command-injection class was deliberately shifted on, received the two highest average weights (0.872 and 0.770 respectively), while the remaining eight, purely nuisance features for the rare class, received weights below 0.6; packet rate, the single most purely benign-vs-common-attack discriminating feature with no rare-class signal, received the lowest average weight of all (0.249). This is the clearest evidence this evaluation offers that the optimizer is doing what the method's name promises: finding and up-weighting the features that actually carry the rare attack's signal, rather than simply fitting noise.

**Figure 3: Mean, over 5 seeds, of the grey wolf optimizer's learned per-feature attention weight. Red bars mark the two features the rare command-injection class was designed to shift.**

**Figure 3** Mean, over 5 seeds, of the grey wolf optimizer's learned per-feature attention weight. Red bars mark the two features the rare command-injection class was designed to shift.



The optimizer's mean tuned hidden-layer width across the five seeds was 208 (close to the upper bound of 220, suggesting the larger end of the searched range was generally preferred), and its mean tuned ridge strength was 0.040, roughly four times the plain ELM's fixed, untuned 0.01, indicating that somewhat heavier regularisation was consistently favoured once the attention vector was also being optimised.

### Per-Seed Variability

The standard deviations in Table 4 are large relative to the means for every rare-class metric, and this is a genuine, informative finding rather than noise to be explained away. Rare-class F1 for the proposed method ranged from 0.161 (its weakest seed) to 0.784 (its strongest) across the five runs; the same seed that was

weakest for the proposed method (rare-class F1 0.161) was also the weakest for Random Forest (0.194) and for the plain ELM, indicating that some random realisations of the synthetic stream are simply harder, for every method, than others, most likely because of chance co-location of the rare class's few dozen test examples near the decision boundary rather than because of any one method's specific weakness. An operator evaluating this method on a real network should expect, and plan monitoring and fallback procedures around, this degree of run-to-run variability rather than treating the five-seed mean as a performance guarantee.

## **Discussion**

### **What the Attention Vector and Quantum-Inspired Initialization Actually Buy**

The comparison that matters most for assessing Shahiduzzaman *et al.*'s method on its own terms is not against Random Forest but against the plain ELM: both share the identical closed-form architecture, so the gap between them isolates the contribution of quantum-inspired initialisation, attention-based feature weighting, and GWO tuning, taken together. That gap was large and consistent across all five seeds (G-mean and rare-class F1 both improved by 45-63% in relative terms), and the mechanism behind it is visible directly in the learned attention profile: the optimizer reliably found and emphasised the two features that actually carried the rare attack's signal. This is a meaningfully different, more interpretable story than simply reporting that a more complex model won; it shows what the added complexity is doing.

### **Competitive With, Not Decisively Better Than, Random Forest**

Against Random Forest, an ensemble method with no attention mechanism, no quantum-inspired anything, and a training cost many times higher than an ELM's single matrix solve, the proposed method was competitive rather than decisively superior: ahead on macro-F1 and G-mean, essentially tied on rare-class F1, and behind on rare-class precision. This is, if anything, a more credible result than a clean sweep would have been: Random Forest is a well-established, carefully engineered baseline for exactly this kind of tabular, imbalanced classification task, and a newly implemented method matching it on balanced metrics while running on a fraction of the training cost (one matrix solve per GWO candidate versus growing 250 full decision trees) is a genuinely useful finding for an operator who needs to refresh a detector frequently on live factory-floor data.

### **Precision-Recall Trade-off Under Severe Imbalance**

The plain ELM's perfect rare-class precision paired with low recall, and the proposed method's lower precision paired with higher recall, reflect a predictable trade-off under 1.5%-prevalence imbalance: a classifier that rarely predicts the rare class will rarely be wrong when it does, but will also miss most true instances. Which side of this trade-off is preferable is an operational, not a purely statistical, decision: a factory operator who treats every rare-class alert as triggering a costly physical inspection may prefer the plain ELM's precision, while one who treats a missed command-injection event as the far costlier failure mode will prefer the proposed method's substantially higher recall, accepting more false alerts to catch more true ones.

## **Conclusion**

This study implemented, from scratch and end to end, the attention-guided, quantum-inspired, grey-wolf-optimizer-tuned extreme learning machine that Shahiduzzaman *et al.* proposed for rare-attack detection in IIoT networks, and evaluated it on a synthetic but deliberately severely imbalanced factory-automation traffic stream against a plain ELM and a Random Forest. The implementation delivered a substantial, consistent improvement over the plain ELM it is built from on every rare-class-sensitive metric, and performance competitive with, though not uniformly better than, a considerably more expensive Random Forest ensemble. Its learned attention vector, inspected directly rather than taken on faith, consistently identified the features that actually carried the rare attack's signal. Its principal operational caveat is a precision-recall trade-off, and

a degree of run-to-run variability, that any operator piloting this method on a real factory-automation network should plan for explicitly rather than read past.

### References

- [1] Shahiduzzaman, M., Sourav, M. S. A., Kabtia, M., Shammah, R. S., Islam, M. A., Chakraborty, P., ... & Adnan, M. (2026). Attention-guided quantum-inspired extreme learning machine with grey wolf optimization for rare-attack detection in IIoT networks. *Discover Artificial Intelligence*.
- [2] Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1-3), 489-501.
- [3] Han, K.-H., & Kim, J.-H. (2002). Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, 6(6), 580-593.
- [4] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, 69, 46-61.
- [5] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- [6] Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access*, 10, 40281-40306.
- [7] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321-357.
- [8] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.
- [9] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- [10] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3), Article e0118432.