International Journal of Science, Strategic Management and Technology
Volume **02** Issue **04** April-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Restus -Real-World AI/ML-Based Phishing Detection and Prevention System

Priyanka Thange[1], Janhavi Jadhav[1], Amit Apte[1], Prof. Priya Godse[2]
[1]BCA, [2]Faculty & Guide [1]Ajeenkya D.Y. Patil University, Charholi Budruk, Pune, India [2]Department of Engineering, [2]Ajeenkya D.Y. Patil University, Pune, India.

https://doi.org/10.55041/ijsmt.v2i4.316

Cite this Article: Thange, P., Jadhav, J. & Apte, A. (2026). Restus -Real-World AI/ML-Based Phishing Detection and Prevention System. International Journal of Science, Strategic Management and Technology, *02*(04). https://doi.org/10.55041/ijsmt.v2i4.316

ABSTRACT:
These days, phishing scams grow sharper, using clever tricks to fool people and slip past standard security checks. Old methods like blocking known hazardous sites or applying fixed rules struggle when facing fresh, changing dangers. To tackle this problem, the study introduces RESTUS—a mix of smart tools powered by artificial intelligence combining CNNs, LSTMs, and LightGBM working together.
One way this setup works involves pulling out features in layers—mixing how things are built, where they come from, and what they contain—to get better results when sorting data. On top of that, RESTUS uses methods that show why decisions happen, helping people understand and rely on its output. The API structure, designed for growth and paired with a responsive front layer, enables live threat spotting. Tests reveal it handles tasks more effectively than older or isolated models ever did, fitting well into today's security demands.

KEYWORDS:
Phishing Detection with Hybrid AI Models Combining CNN, LSTM, and LightGBM Using Explainable AI and RESTUS in Cybersecurity

INTRODUCTION:
Out here, where tech moves fast, danger online shifts just as quick. Phishing sticks around though - still a major problem despite everything else showing up. Someone pretending to be your bank might send an email; others fake sites that look like Facebook or work portals. Tricking people is the goal, getting passwords or data through lies built to feel safe.
Out of nowhere, phishing grew sharper because so many people now bank, chat, and save files online. That shift gave hackers extra reason to level up - using bots and smart software to mimic real messages almost perfectly.
Older ways to catch phishing, like blacklists, work only if the bad websites were already flagged. When new fake sites pop up, these methods miss them completely. Rules that spot set patterns also fall short - small tweaks by scammers break their detection. Attackers change a letter here or shuffle code there, and the system looks away. What worked yesterday often fails today without warning. Each tweak outsmarts filters built for yesterday's tricks.
Learning from data helps systems adapt, yet most still depend on just one method, weakening how well they spot phishing tricks. Though strong, some smart models hide how they reach conclusions, leaving people confused about why a site gets flagged. Rules alone fall short, so pattern recognition steps in - but narrow approaches miss sneaky variations that evolve constantly. Understanding each judgment matters, especially when hidden layers make choices without showing clear reasons behind them.
Out of many AI methods comes RESTUS - a mix built to catch phishing more reliably. Not just one trick powers it; instead, layers stack together: CNN spots form, LSTM tracks flow, LightGBM handles decision paths. Real time keeps pace without lagging behind. Structure meets sequence inside its core. Efficiency stays high even when complexity rises.

Clarity does not get lost along the way. Learning adapts as new patterns appear. Speed holds steady under pressure. Together, pieces work where others fall short.

II. LITERATURE REVIEW:

Now things shift when machines start learning how phishing gets caught. At first, people leaned on fixed rules plus number tricks - these often failed because fresh scams slipped right past them.

Learning machines brought a real step forward, training on tagged examples instead. Simple methods like Logistic Regression or Decision Trees came first - easy to grasp, clear in logic. Yet when data grew messy or packed with features, they frequently fell short.

Out of different approaches, using groups of tree-like predictors - like Random Forest - tends to work better because it smooths out errors through variety. Instead of relying on just one model, stacking several together handles messy inputs more reliably, which often leads to sharper results [10].

Deep learning moves forward, pushing complex tools like CNNs into spotting phishing tricks. These models catch layout clues inside links or web pages quite well. On the flip side, LSTMs handle streams of info - say, words in emails or parts of a URL - one piece at a time [7], [13].

One study looked at mixed methods, linking different algorithms to get better results. A system using CNN along with gradient boosting showed stronger outcomes in spotting phishing attacks fast [12]. Email texts are now examined by machines that understand language, catching odd word choices tied to scams [2].

Even with progress, problems still pop up here and there. Some tools give no clear reason for their choices, leaving users guessing what happened behind the scenes. On top of that, results tend to slip when models face fresh data, showing they struggle beyond familiar ground [9]. All this points toward needing something smarter - something that bends instead of breaks under change.

III. PROBLEM STATMENT:

Even with advances in catching phishing scams, current tools struggle when it comes to spotting fresh threats they've never seen before. A major hurdle shows up right away - these systems often miss brand-new phishing attempts because there's no prior record to match against.

One big problem? It's how most machine learning setups still depend on hand-crafted features. These need deep knowledge of the field, yet often miss hidden patterns in information. When scam methods change - detection built on fixed traits starts lagging behind. Older signals lose power, simply because they were made by people guessing what matters.

Instant handling matters just as much. Some current tools work only after the fact, failing to catch threats the moment they appear. Because of that delay, tasks like guarding web browsers or sorting emails can't rely on them when speed is essential.

When systems grow, keeping up with heavy data loads gets tricky fast. Hard to see inside complex models means people often doubt their choices, especially when sorting out tough cases.

What these issues show points toward something clear - a way of working must shift when conditions change, stay precise under pressure, grow without breaking, yet remain open to inspection.

IV. PROPOSED SYSTEM:

A. System Overview

Restus works as a set of connected levels, joining screen design, calculation core, and decision rules into a single system. At the top, users enter website addresses or written messages using a clear layout built for smooth typing. Beneath that, an adaptable structure takes every new job and moves it along defined tracks. Rather than isolated pieces doing their own thing, all sections link together so steps follow naturally from beginning to end. Right away, things start on the outside before slipping fast into complex calculations. One part works alone, though meaning comes once tied to what's nearby. A shift happens up front, then layers unfold quickly beneath. Alone, each element does its job - clarity appears only when connections form.

At its core lies an artificial intelligence system designed to catch phishing using various techniques at once. Next to this, data storage keeps track of people's information, results from scans, and logs showing what the system has done. Since

the setup divides tasks among separate sections, changes in demand do not cause disruption. When pressure increases, performance stays steady due to seamless connections between components.

B. Data Processing Pipeline:

Right away, RESTUS grabs rough stuff - web addresses or chat snippets - exactly how users send them. Then cleaning kicks in: odd symbols vanish, spelling slips are corrected, layout becomes consistent. Once neatness is done, only then do fragments shift toward meaningful predictions. Step follows step without leaps, each stage setting up the next. Nothing jumps forward until prior work settles.

Later on comes sorting - some parts stand out more than others. Layout patterns begin appearing, hints about topics start forming, even word habits join in. Once prepared, a blend of methods picks them up. Each bit gets studied closely before finding its place.

Here, whoever acts sees what happens - along with why every move was taken. One piece at a time, the process stays clean, focused, never losing its edge. Each turn unfolds plainly, keeping clarity front and center without slipping.

C. Feature Engineering:

Feature shaping matters a lot behind RESTUS success. Inputs built carefully tend to run smoother. Not every type appears equally though. Most seen? Exactly three stand out clearly. First up are words, along with how they appear. The location of information plays a role just as much. What the content truly means sits beside that. One kind shapes forecasts in ways another does not.

Right away, the way a URL looks gives clues - length, strange characters, suspicious terms hidden within. These features stand out because they're quick to analyze, spotting dangers on the move [5].

Out of nowhere, details pop from the hosting background tied to a site. When the domain age seems off, alongside oddities in DNS records, suspicion grows. Name servers revealing mismatched info can speak volumes. If SSL certificates appear irregular, it adds weight. New signups paired with strange activity tend to stand out. These pieces together speed up spotting ones that feel wrong.

Webpage structure and behavior define what we call content-based features. Elements such as form layouts, snippets of code, or pieces written in HTML fall into this group. As these details grow clearer, the demand for stronger processing rises just the same [7].

What makes RESTUS stand out is how it combines these elements, giving a clearer view of the input. Because of this blend, accuracy improves, especially when things get unpredictable. The system handles varied situations well, thanks to its layered approach.

D. Hybrid AI Model:

Restus stands apart by blending AI strategies to sharpen results. Not stuck on a single method, it layers multiple approaches quietly underneath. One core piece uses CNNs - they detect forms and arrangements within data streams. Thanks to this step, patterns common in scams get flagged early. Subtle structural hints turn into signals, shaping choices even when no obvious markers exist.

Step by step pattern tracking makes LSTM good with sequences, particularly in URLs and short bits of text. What stands out is how it catches clever phishing tries hiding in tiny contextual clues.

From the CNN and LSTM flows a series of signals - LightGBM collects these, molds them into choices. Speed holds firm, yet accuracy doesn't slip. The process moves quickly, still keeps its edge.

Something stronger takes shape when one system passes results to the next. Not every piece needs perfect timing to work right. Because it adapts through varied perspectives, breakdowns happen less often. Connections between methods add resilience more than any single approach could. Even if details change a little, the result holds firm. How things fit together matters most in keeping performance reliable.

E. Explainable AI Integration:

Most tools keep their logic hidden. RESTUS turns that upside down through clear AI techniques. Step by step, it reveals how judgments form within the system. Clarity appears when people watch which signals trigger a phishing alert.

Guessing fades once the process unfolds before the eyes. Trust grows quietly as layers of confusion drop off. Reasoned insight moves forward without clutter or guesswork blocking sight.
Out front, placing what counts most lets folks spot how conclusions form - suddenly, things seem less hidden. That clarity? It nudges trust upward when shielding systems online, especially when reasons behind decisions carry weight.

V. METHODOLOGY:

This research creates a clever system to catch false messages using blended artificial intelligence techniques. Fast in operation, precise in results, flexible when adjusting, while offering transparent decision logic. The process moves forward one piece at a time - pulling unprocessed data early on, after which tidying it with care. Then structuring vital features unfolds, before building the forecasting mechanism takes place. Practice runs teach the engine how to spot patterns before it faces strict tests. Because every step links closely, outcomes stay reliable across all stages.

A. Research Design:

Out in the wild, data shapes how tests are made to catch phony sites with clever code. Not stuck on one trick, it combines moves that boost each other's strength. Designed to shift as needed, the system blends learning types like a mixtape of minds. When one piece misses something, another steps in to cover it. Harder to fool than solo models, this team-up sharpens accuracy by letting systems back each other up.
The methodology emphasizes:
1. Real-world dataset usage.
2. Multi-feature analysis.
3. Hybrid model integration.
4. Explainability and transparency.

Starting at the beginning helps keep ideas clear when used outside theory. Still, it must survive questions from experts too.

B. Dataset Collection and Integration:

Surprises down the road? Easier to manage when inputs come from more than one place. Size isn't the only player - shape counts just as much. Learning depends heavily on where the material comes from. Pieces gathered widely build tougher results.
Some of the items found within the collection include these parts

1. Phishing Data:

Out of public records and real-time feeds, phishing URLs along with suspicious emails are pulled in. These samples reveal varied tactics - one might grab login details, while another copies a familiar website. Messages dressed up as trusted contacts also turn up now and then.

2. Legitimate Data:

Real web addresses and messages get added first, that is how fairness begins. Everyday usage shows up next to trusted websites, mixed together. The system learns to separate unsafe material from safe through contrast instead of fixed guidelines. Differences stand out clearly when studied closely like this.

3. Email-Based Phishing Data:

Messages meant to trick pop up right next to ones trying to steal passwords in crowded inboxes. That forces experts to study patterns between phrases found in various frauds. At times, scam letters go after just one person - those need deeper scrutiny. The way sentences form changes based on the deception applied. Hidden goals behind false wording tend to surface through small hints around them.

4. Synthetic and adversarial data:

Out of nowhere, false messages show up, quietly shifting into different forms. Each twist becomes a new trial, slipping past what the system already knows. Because of this, responses must adjust, shaped by differences it hasn't seen before. Surprise sneaks in through small edits - learning happens when nothing stays fixed.

Out of many sources, information blends into something balanced - still real, like daily life - a key part when catching scams sharp enough [8].

C. Data Preprocessing:

Once the details come together, sorting kicks off right away. Right at that point things shift slightly, just enough to fit each part where it needs to be.

1. Noise Removal:

Bits such as tracking codes, strange tags, or pointless symbols are removed. Less mess means smoother operation because there's less junk in the way.

2. URL Normalization:

Out of nothing, a URL splits apart - protocol first, then destination, path tagging along behind. This shape stays fixed, each piece slotted just so, making sure mismatched entries never slip through.

3. Text Cleaning:

First up, any messy parts vanish from the email snippet - uppercase characters disappear completely. Then, everyday throwaway words are pulled out, leaving just the stuff that matters. Next, pointless punctuation marks? Erased without a trace. When the dust settles, the message sits clear, ready to be examined properly down the line.

4. Tokenization:

The cleaned data is converted into tokens:

1. Character-level tokens for deep learning models.
2. Word Tokens for NLP Analysis.

5. Feature Scaling:

Scaling values evenly stops one number from drowning out another while the system learns. That smooths how adjustments happen, pushing performance ahead with less struggle.

First thing, cleaning the data makes sure everything moves smoothly ahead. Step by step, rough pieces turn into something structured and clean. Once holes are patched and clutter fades, patterns start showing in order. What emerges fits together, ready to reveal what matters.

D. Feature Engineering:

Most important thing here? Shaping those features. That step decides how clearly the model sees connections. From every direction, RESTUS gathers clues all at once.

1. Lexical Features:

Out of sight? Not here. Web addresses split into chunks, each one spilling a detail just by sitting where it does. Structure decides display - no extras, only position speaks. What appears depends entirely on arrangement, exposed through order alone

1. How long the web address is.
2. How many symbols count as special ones.
3. Presence of suspicious keywords.

Faster processing is particularly effective when identifying word patterns in real-time.

2. Host-Based Features:

Information on the domain and hosting setup comes through these characteristics

1. Domain age.

2. DNS records.
3. SSL certificate validity.

Patterns that seem off tend to appear on new or questionable sites - common when scams are underway [4].

3. Content-Based Features:

Out of the way pages are put together, people start to see what's going on. Just like spacing, the way they respond counts. What comes after often depends on how it's stacked. Over days, their moves begin to repeat. Connections? They shift outcomes more than expected. A shape might guide how it's used, quietly. How something looks can decide what happens next, little by little

1. HTML elements
2. JavaScript usage
3. Form actions and redirects

Still, spotting these signs reveals a clearer picture of phishing methods [7].

4. Feature Fusion:

One complete set forms when each separate part joins back. As pieces link up, the system catches broader patterns in fraud tries - boosting its results across the board.

E. Model Development:

A twist of thinking machines powers RESTUS - CNN spots shapes in images, yet LSTM follows changes as they unfold across moments. LightGBM slips in smart choices, weaving through the rest.

1. CNN (Feature Extraction):

Out of the blue, a system that loves finding patterns dives into how web addresses are built plus what shows up on pages. Skipping the need for manually written guidelines, it grabs important features by itself. Though quiet about how it learns, it works without help once started.

2. LSTM (Sequential Analysis):

Every fresh start changes the pattern just enough to keep things unpredictable. When dealing with chunks of data - say, a web link or an email line - LSTM steps in. Since what comes before shapes what follows, it keeps tabs on word links even when they're far apart. Sneaky patterns in phishing tries? It picks up on those. Stuff buried deep in long texts shows up clearly here, while older ways walk right past.

3. LightGBM (Classification):

Here comes the final stage - LightGBM takes charge of sorting outcomes. Signals that traveled separately through CNN and LSTM now merge into one path. Performance holds steady, yet processing flies fast. Quickness joins careful detail without dragging pace.

4. Hybrid Model Integration:

Front and center, CNN outputs blend with what the LSTM brings, then move toward LightGBM. This way of linking lets strengths from each model play out - lifting performance beyond either on its own. Not just stacking, but syncing their roles.

F. Model Training:

Out of clear goals comes learning, built slowly on structured practice. Each new example adds a piece - like stacking bricks one by one. Responses grow sharper, shaped by repeating just one pattern at first.

1. Training Strategy:

The training process involves:

1. Training CNN and LSTM models.
2. Feature extraction.
3. Final Training With Lightgbm.

2. Loss Function:
Wrong guesses by a model are measured with binary classification loss. Through this measure, the system learns to shift its settings gradually. Perfection isn't the first goal - small updates follow from errors instead. Each adjustment leans on what the score reveals. Small fixes point the way ahead. With every try, understanding grows a little more.

3. Optimization:
Off to a different start, techniques such as Adam adapt their pace while learning. Because of these shifts, systems find solutions more quickly. Rather than holding firm, they react as data moves through them. When necessary, strides grow shorter or longer - making movement steadier overall. Midway through, changes begin shaping each run on their own. Progress finds its rhythm more smoothly when left to adjust freely.

4. Regularization
Early halting of training, or tossing out random pieces during learning, stops a model from sticking too tight to what it sees. Because of that, it handles fresh cases more smoothly - less stuck on details it picked up by accident.

5. Cross-Validation
Every time the data divides, K-fold cross-validation measures how reliably the model behaves. Not just once but several times, the cycle runs to check stability. As folds take turns being tested, patterns emerge - do scores stay close or jump around? Once per round, each group serves as validation, giving every sample equal chance. Fairness comes from rotating roles, so no segment gets special treatment. The whole approach spreads testing weight equally across all parts.

G. Model Evaluation
Checking the model's performance often involves using standard evaluation methods:
1. Accuracy: Overall correctness
2. Correct Positive Predictions
3. Recall Detecting Phishing

Precision doesn't always see eye to eye with recall - odd pair, really. Yet, they manage to find a balance. The F1-score watches that meeting closely.
Additional evaluation includes:
1. ROC curve analysis.
2. Precision-recall analysis.
3. Confusion matrix.

Up close, small things reveal just what shape the whole setup is in.

H. System Implementation Workflow:
The RESTUS system follows a real-time workflow:
1. User Provides URL or Email.
2. The backend validates input.
3. Data is preprocessed.
4. Features are extracted.

Together, handling both pieces changes the way incoming stuff gets managed:
1. A prediction is generated
2. Explainable AI shows how decisions are made
3. User gets the outcome back

Quick spotting of phishing relies on seamless step links. Checks flow without hiccups when pace stays even. Mistakes slip in less often that way.

I. Explainable AI Integration:

What makes sense now? RESTUS shows the steps behind each decision. Not left wondering, users spot exactly which facts tipped the balance. One number, one detail—each gets its moment under a clear light. You notice what pulled strongest because the path unfolds like footprints. Seeing weight means seeing why. What you notice builds confidence over time. Not tricks—simply knowing what happens out of view.

This improves:

1. Transparency
2. Trust
3. Debugging capability

Decisions in cybersecurity carry weight because their reasons shape what happens next. Not every move is clear until you see the thinking behind it.

J. Deployment Methodology:

Grow it later, no problem. Handles extra work without lagging behind. Pieces join up without hiccups when adding on. Runs just as fast even when busy. Changes fit right in, whenever you need them

1. Backend API with FastAPI.
2. Frontend Interface Built With React.
3. Database for data storage.

Running on remote servers lets it grow easily

That's why RESTUS handles real-time traffic while juggling large amounts of data—no lag involved.

V. IMPLEMENTATION

Running the RESTUS setup leans on fresh web tools tied together with smart algorithms, boosting speed and live response across growing demands. Built like puzzle pieces fitting together, every section grows apart then links up - shaping one sharp tool that spots fake messages.

Behind RESTUS runs software built with FastAPI, a quick Python tool praised for moving fast and managing tasks at once. Because it works ahead of time, handling what users ask for happens without waiting long. This setup takes in messages from people, sorts through information, then sends back answers before delays pile up. Working step by step while juggling many pieces helps catch fake attempts instantly, even when crowds hit the system. Security stays tight through special keys made with JWT, locking out unwanted access during exchanges.

Inside the backend lives an AI setup built right into the core. This one uses a mix of CNN, then LSTM, followed by LightGBM working together. Once someone sends a link or message, things start moving fast. First comes cleanup of data, then pulling out key details, after that prediction happens swiftly. Models fire up quickly thanks to smart loading tricks. Speed matters here - results come back almost instantly.

Built with React, the interface handles dynamic layouts through efficient updates behind the scenes. A clear dashboard appears right away, welcoming visitors who paste links or email text into fields. Instead of clutter, it shows findings plainly - labels appear beside breakdowns made visible by Explainable AI tools. Speed matters here; each click responds without delay thanks to how elements load only when needed. Smooth interactions come naturally since parts redraw themselves only when something actually changes.

Storing and managing data happens through a relational setup, like MySQL. Information about users lives inside alongside their scan records, outcomes, their activity traces. Indexes get set up carefully; tables follow logical splitting rules - this keeps lookups fast and accuracy intact. Reviewing earlier scans becomes possible because of how things are arranged here. Watchdog roles check efficiency trends thanks to consistent layout choices.

Starting with flexibility, the setup works well across cloud environments for growth-ready use. When demand rises, traffic spreads smartly through balanced routing plus backend distribution tricks. Each piece stands apart in structure, so updates happen solo - no chain reactions. Built this way, scaling fits neatly around change, not against it.

VI. RESULTS AND DISCUSSION:

Restus gets tested thoroughly with common measures like accuracy, yet also looks at how often it catches real scams. Not just spotting fake sites right, but not missing actual threats matters too - precision plus recall shape that view. Instead of one number telling everything, the F1-score blends both sides into a clearer picture. Each result helps see where the system holds up, even when patterns shift slightly.

What stands out is how well the hybrid approach works compared to older methods or deep learning alone. Part of why it does better lies in pulling together what CNNs, LSTMs, and LightGBM do best. Features in website structure and links get picked up clearly thanks to CNNs. Following order in words and phrases? That's where LSTM steps in. Then LightGBM takes those insights and blends them smartly for sharper predictions.

What stands out in the data is fewer mistakes spotting fake sites. Missing an attack becomes less likely when errors on that front drop. Because it rarely labels safe pages as dangerous, trust stays high. This method manages both sides well - catching scams without scaring users unnecessarily. Real use cases benefit when systems behave like this.

When explainable AI gets added, systems become easier to use. Rather than just saying something is phishing, they explain the reason behind that decision. Take strange words in messages - those get flagged clearly. Long web addresses stand out too, along with domains created very recently. People feel more confident using tools like this. Experts studying threats also gain better clarity on how attacks are built.

Besides being accurate, the system handles tasks quickly, measured by response time and volume capacity. Response times fall under one second, often landing between half a second and eight-tenths of a second. Because it reacts so fast, it works well inside tools like web browser add-ons or programs that sort incoming messages.

Stability sticks around even when pressure builds up, tests show. When loads spike, the setup keeps working - no hiccups, just steady flow. Behind the scenes, tasks move independently, handling bursts without slowing down. Strength grows where speed meets demand, quietly proving it can scale. Real-world chaos? Restus proves itself through solid outcomes, showing strong precision without sacrificing speed. What stands out is how smoothly it fits into real-world use, beating older methods at their own game. Not only does it detect scams well, but it also works faster than most alternatives. Its edge comes from balancing smart design with everyday function. In practice, this means fewer mistakes and less effort for users.

VII. CONCLUSION

Out there among digital threats, something new shows up - RESTUS, built different, mixing AI styles on purpose. Not stuck like older systems, it pulls together pieces: CNN watches patterns, LSTM tracks sequences, LightGBM weighs decisions - all working without rigid lines between them. Instead of betting on just one method, this blend adapts where others freeze up. Each part adds weight only when needed, shifting balance quietly behind the scenes.

Not only does blending various models help uncover diverse aspects of phishing data - like structure, sequence, and context - it also builds a fuller picture. Detection becomes sharper, more reliable, when these layers come together.

What stands out here is how this work brings **Explainable AI** into the mix, opening up the black box just enough to build confidence. When it comes to cybersecurity, knowing why a system makes certain choices matters deeply - RESTUS meets that need without overcomplicating things.

Running on tools like FastAPI and React, this setup keeps things fast even under heavy load. What helps it work smoothly is how it manages huge amounts of information without slowing down. Real situations demand speed, that is why response times stay sharp. Built to grow, it adapts as demands increase over time. Heavy usage does not break rhythm, thanks to its responsive core design.

What stands out is how RESTUS brings together precision and speed without sacrificing clarity. Instead of just spotting scams better, it builds confidence through straightforward design. Performance gains come hand in hand with smoother interactions. Scalability fits naturally into the setup, avoiding complexity. Clear outcomes emerge not by chance, yet through deliberate structure. Trust grows because users see what happens behind decisions.

VIII. FUTURE WORK:

Even so, RESTUS shows solid results, yet a few paths remain open to push it ahead through deeper study. Still, its current success doesn't close the door on new tests or fresh questions worth exploring. Now and then, small gaps appear - inviting closer looks, different angles. Every strength hides a next step, after all. Progress rarely stops where it stands.

Starting fresh each moment, it might pull insights from how **reinforcement learning** works - adjusting on the fly when faced with unseen phishing tricks. Instead of starting over completely, improvement happens bit by bit through steady exposure and response. The model grows sharper, not from overhaul but repetition, shaped slowly like water over stone.

One key challenge lies in spotting **phishing** attempts made by AI. When cybercriminals start using smart software to craft convincing fake messages, defenses need sharper tools to catch these tricks. Detection has to evolve just as fast as the deception.

Out in the open, RESTUS might find a home on **edge** hardware. Right there on your gadget, it crunches data fast - slashing delays while guarding personal info tighter. Think web browsers, watching threats without phoning home.

Fed **learning** slips into the mix, quietly boosting how safe info stays. Training happens on separate gadgets, never moving the original files - this sidesteps leaks pretty well. Performance doesn't drag either, even with everything staying put.

One step ahead could mean digging into richer data pools, while stretching model skills beyond familiar zones. That kind of stretch helps maintain steady results, whether facing new settings or shifting threat patterns.

REFERENCES

[1] B. K. Sospeter, A. Nugroho, and R. Hidayat,
"AI-Based Phishing Attack Detection and Prevention Using Natural Language Processing,"
Proc. IC-ITECHS, 2024.
[2] O. A. Lamina, A. O. Adewumi, and M. A. Adebayo,
"AI-Powered Phishing Detection and Prevention: A Real-World Perspective,"
Path of Science, vol. 11, no. 2, pp. 101–115, 2025.
[3] H. Jabbar and S. Al-Janabi,
"An AI-Driven Reinforcement Learning Framework for Phishing Detection,"
Big Data and Cognitive Computing, vol. 5, no. 2, 2025.
[4] A. Agrawal, S. Verma, and R. Singh,
"Application of Machine Learning for Real-Time Phishing Attack Detection,"
JETIA, vol. 11, no. 1, pp. 45–52, 2025.
[5] S. Rehman, M. A. Khan, and T. Hussain,
"Real-Time Phishing URL Detection Using Machine Learning,"
Engineering Proceedings, vol. 107, no. 1, 2025.
[6] P. Shelke, R. Patil, and S. Pawar,
"Defending Future Phishing Website Assaults Using Machine Learning Algorithms,"
IJISAE, vol. 12, no. 3, pp. 233–241, 2024.
[7] R. S. Dhole, A. Kulkarni, and S. Jadhav,
"AI-Powered Phishing Attack Detection Using Deep Learning,"
IJARCCE, vol. 14, no. 1, 2025.
[8] M. A. Rahman et al.,
"Artificial Intelligence Techniques for Phishing Detection: A Review,"
Frontiers in Artificial Intelligence, vol. 8, 2025.
[9] Y. Chen, L. Zhang, and D. Wang,
"Feature Reliability in Phishing URL Detection,"
Proc. ACM AsiaCCS, 2025.
[10] A. Al-Qasmi, M. Al-Shabi, and H. Al-Balushi,
"Machine Learning-Based Phishing Detection Using Random Forest,"
IJISAE, vol. 12, no. 4, 2024.

[11] Z. Alshingiti et al.,
"A Deep Learning-Based Phishing Detection System Using CNN and LSTM,"
Electronics, 2023.
[12] M. Elberri et al.,
"A Cyber Defense System Against Phishing Attacks Using CNN-LSTM Hybrid Model,"
International Journal of Information Security, 2024.
[13] CNN + LightGBM Ensemble Model (Research-Based Concept)
Used for high-speed classification and production systems.
[14] AntiPhishStack (LSTM-Based Model)
Demonstrates robustness against evolving phishing patterns.
[15] CNN-LSTM + XAI Phishing Detection Framework
[16] R. Dubey et al.,
"Phishing Detection Using CNN + LightGBM Ensemble,"
arXiv, 2025.
[17] S. Yerima and M. Alzaylaee,
"High Accuracy Phishing Detection Using CNN,"
arXiv, 2020.
[18] S. Baskota,
"Phishing URL Detection Using Bi-LSTM,"
arXiv, 2025.
[19] S. Abdelnabi et al.,
"VisualPhishNet: Zero-Day Phishing Detection Using CNN,"
arXiv, 2019.
[20] R. Ayyasamy et al.,
"Hybrid Deep Learning Framework Using LSTM,"
Scientific Reports, 2025.