ChatGPT

VEINN: A Vector-Encryted Invertible Neural Network for Post-Quantum Encryption

Abstract: In the emerging post-quantum era, traditional cryptosystems (e.g. RSA/ECC) are threatened by quantum algorithms like Shor's factoring and Grover's key search [1] [2]. VEINN (Vector Encrypted Invertible Neural Network) proposes a novel *machine-learning–inspired* layer to *augment* classical encryption: it maps plaintext into a high-dimensional real-valued vector space, then applies a secret key–dependent invertible neural network (INN) transform. This *one-way* nonlinear mapping (realized by stacked affine coupling layers) scrambles the data in a way that is reversible only with the correct key. In effect VEINN acts as a large, pseudo-random permutation (block cipher) driven by a neural network, with the intent of "hardening" existing ciphers against quantum attacks. We analyze VEINN's design and novelty (high-dimensional nonlinear embedding, key-derived noise, configurable layers), compare it to standard cryptographic approaches, and discuss its *polymorphic* extensibility – e.g. how one could integrate or "wrap" VEINN with AES or lattice-based schemes. While purely experimental and without formal security proofs, this approach opens a new direction: layering reversible neural-net mixing to increase the complexity seen by a quantum adversary.

Introduction

Quantum computers challenge classical cryptography: Shor's algorithm can factor large primes and break RSA/ECC, while Grover's algorithm can roughly halve symmetric key strengths [1] [2]. In response, NIST and others have standardized post-quantum algorithms (e.g. CRYSTALS-Kyber, CRYSTALS-Dilithium, SPHINCS+, FALCON) based on lattices, codes, or hash problems [3] [1]. These rely on structured algebraic hardness. VEINN proposes a *different* tactic: rather than a new hard mathematical problem, it embeds data into a continuous, high-dimensional vector space and applies key-dependent *neural* transformations, hoping that the resultant complexity is resistant to known quantum attacks. In other words, VEINN is a nonlinear *embedding layer* that can be composed with existing encryption, analogous to adding an extra "chaotic mixer" under a shared key.

At a high level, VEINN works as follows: - The plaintext (bits or bytes) is first encoded into a real-valued vector in $[-1,1]^n$. (For example, each block of data is normalized or mapped linearly to floats.)
- A secret key is used to seed a family of *invertible* neural network layers. Typically, these layers are **affine coupling layers** (as in RealNVP/GLOW architectures) which implement a bijective mapping. Each layer splits the vector into two halves, transforms one half by a small neural subnet (whose weights or biases are keyed), and combines them. Stacking many such layers yields a highly nonlinear, reversible map [4].
- Optionally, a small amount of deterministic, key-derived "noise" or offsets can be added in each layer to ensure exact reversibility (for decryption) [4] [5]. This noise is derived from the key so that decryption can subtract it out.
- The output of the final INN is taken as the ciphertext vector; in practice it might be quantized or stored alongside the key state.

Key features of VEINN include: - **High-dimensional vectorization:** Plaintexts are mapped into real-valued vectors (each coordinate in $[-1, 1]$), preserving more "room" for transformation than simple bitwise XOR. This creates many degrees of freedom [6].
- **Invertible neural network layers:** A sequence of affine coupling (reversible) layers provides *nonlinear, reversible* diffusion. Crucially, the overall mapping is bijective, ensuring that decryption (the inverse network) exactly recovers the plaintext when the key is correct [4].
- **Key-derived noise:** Small deterministic perturbations (derived from the secret key) can be injected to break linear correlations and guarantee invertibility, yet can be canceled by the legitimate decryptor [5].
- **Configurable depth:** Users can vary the number of layers or the internal network size to trade off performance and complexity [6].

The core motivation is that quantum algorithms (like Grover) operate over structured domains (e.g. keys, algebraic groups) and only achieve quadratic or exponential speedups over brute force. By embedding data in a high-dimensional continuous space with a complex neural mapping, VEINN hopes that attacking the cipher requires *searching a vastly larger, unstructured space*. This idea resembles recent research that composes traditional encryption with neural layers to demand large quantum memory resources [2]. In particular, Węgrzynek & Topa (2023) propose composing any symmetric cipher with a wide neural network so that decrypting (without keys) requires having huge memory on the quantum computer; since qubit counts grow slowly, this impedes Grover's attack [2]. VEINN is another instantiation of this concept: the INN adds non-algebraic "noise" and entropy to the cipher.

Design and Novelty

VEINN is non-trivial and novel for several reasons:

1. **Machine-learning meets cryptography:** Traditional block ciphers use fixed algebraic or S-box operations. VEINN instead uses a *trained/parameterized* neural mapping as the core permutation. Invertible neural networks (e.g. RealNVP [4]) are well-studied for density estimation and style transfer, but using them directly as a cryptographic primitive is unusual. The reversible property (no information loss) is key: it means the neural net can act like a permutation of $\mathbb{R}^n$ keyed by parameters. Designing cryptographic transformations from neural nets is relatively unexplored and not based on a standard hardness assumption, making it a novel (though risky) approach.

2. **High-dimensional chaos:** By converting the plaintext into a high-dimensional float vector, VEINN breaks it out of the usual bitwise linear domain. The invertible network layers mix these coordinates in nonlinear ways (e.g. using learned scaling and shifting functions). Such high-dimensional, highly entropic transformations may be difficult for an attacker to reverse without the key, since there is no simple algebraic structure to exploit. In essence, VEINN attempts to "amplify entropy" by mixing bits thoroughly, akin to a complex diffusion layer [7]. Unlike lattice-based PQC which still has algebraic traps, VEINN's mapping is intentionally chaotic.

3. **Adaptivity and parameterization:** VEINN's algorithm is polymorphic in that it can adopt many forms depending on the key. Changing the secret key can reconfigure the entire network (its layer weights, biases, noise), so each instance is effectively a different cipher. This dynamic behavior resembles *polymorphic encryption* (where the algorithm changes shape per key) [1]. In practice, VEINN allows multiple customization points: different numbers of layers, different neural subnet designs, or alternate coupling strategies. One could even insert classical cipher rounds (e.g. an AES

round) as a sub-component of a layer. This extensibility means VEINN is not a fixed cipher but a *framework*: one can **wrap existing block ciphers or PQC schemes inside a VEINN pipeline**, providing an additional nonlinear mixing step [8]. For example, one might first encrypt with AES, then feed the ciphertext through the VEINN transform (or vice versa). This sort of "hybrid" layering is analogous to the TRIP proposal, which emphasizes that neural-net layers can compose with standard cryptosystems [8].

4. **Quantum-hardening perspective:** The primary novelty claim is that embedding encryption in a continuous, high-dimensional vector space *could* raise the bar for quantum attackers. While not based on a traditional hard problem, the hope is that without knowledge of the key-dependent mapping (the neural network parameters), an attacker faces a complex inversion problem in a real space. In particular, generic quantum search (Grover) offers at best a quadratic speedup. If the effective keyspace (the space of possible invertible neural maps) is extremely large, then even a quantum adversary might struggle. As Węgrzynek & Topa observe, enforcing that decryption requires large memory or computational resources can hedge against Grover's algorithm [2]. VEINN can be seen as attempting to do this by creating a "wide net" of transformation, although no formal proof exists.

In summary, VEINN's novelty lies in its *architecture fusion*: it is a symmetric, key-based cipher realized via an invertible neural network, producing a permutation of message vectors. This is a departure from both classic ciphers (which use fixed arithmetic/GF operations) and from fully algebraic PQC. The design is highly non-trivial because reasoning about its security requires new analysis tools (for example, studying how INNs behave on discrete/quantized inputs, or how an attacker might learn/approximate the inverse mapping).

Architecture Details

The encryption (and decryption) pipeline of VEINN can be described as follows:

- **Message Vectorization:** Given a plaintext (e.g. a byte string or image), it is mapped into a real vector $x \in [-1,1]^n$. This could be as simple as treating each byte as an integer in $[0,255]$ and scaling to $[-1,1]$, or using a fixed linear layer. The dimension n is a parameter – larger n gives more space for mixing.
- **Key generation:** The secret key may be a random seed or a vector itself. From this key, all weight matrices, biases, or noise values used in the INN layers are derived deterministically (via a pseudorandom generator or key schedule). This ensures that encryption and decryption use the *same* invertible transformation when the same key is provided.
- **Invertible Neural Network (INN) Layers:** The core cryptographic transformation is a composition of k invertible layers. A common choice is an *affine coupling layer*: split the input vector into two halves (u,v) ; compute $u' = u$ and $v' = v \odot \exp(s(u)) + t(u)$, where s,t are neural networks (keyed by the key), and $\exp$ is elementwise exponent. Then swap roles of halves for the next layer. Each such layer is bijective if s and t are well-defined. By stacking many layers with alternating partitions, one obtains a highly nonlinear invertible mapping F . In VEINN, the number of layers and their internal structure is configurable [6].
- **Controlled Noise or Offset:** Optionally, a small offset (derived from the key) may be added to the data either as a separate step or within each layer, to act like a secret mask. Because this offset is

deterministic given the key, it can be removed in reverse during decryption. The purpose of this noise is to break any remaining linearity and ensure uniqueness of the mapping, while not breaking invertibility [5].
- **Output:** The final vector $y = F(x)$ is the ciphertext. In a practical system, one might quantize y to a fixed precision or directly store it as floating-point vector; the key needed to reverse F must be stored securely. Decryption simply applies F^{-1} (the inverse network, layer by layer in reverse) to y, subtracts the offsets, and reconstructs the original plaintext vector x.

In implementation, the key can be saved to a file along with the ciphertext, as seen in the current VEINN prototype [9]. The CLI allows arbitrary numbers of layers to be tested. Experimental trials ("parameter sweeps") can measure reconstruction error for various n, k. The developers report that perfect decryption is achieved (by design), but "minor reconstruction drift" can be tolerated for security [10] – implying some small floating-point noise may be acceptable and even desirable to an extent.

Key Insights (Bullet List): The approach is characterized by:
- Vectorizing plaintext to a continuous space to allow nonlinear transformations [6].
- Using multiple *affine coupling layers* (an invertible DNN design) for one-way, key-dependent mixing [4].
- Injecting deterministic, key-derived noise to ensure invertibility and prevent trivial linear attacks [5].
- Treating the resulting scheme as a *configurable, polymorphic cipher* that can wrap or augment existing algorithms [8].

These components differentiate VEINN from conventional ciphers and even from other "neural cryptography" concepts, which often focus on neural key exchange or using neural networks to analyze ciphers (see e.g. Sec. 2 in [11]). VEINN is instead a direct encryption primitive built from a bijective network.

Comparison to Standard Cryptographic Approaches

From a cryptographic perspective, VEINN is most analogous to a large-block symmetric cipher (a huge S-box or P-box network) keyed by a secret. It differs significantly from popular ciphers:

- **Classical symmetric ciphers (AES, ChaCha, etc.):** These rely on fixed linear and nonlinear algebraic operations (XORs, modular additions, small S-boxes) arranged in rounds. They are extensively analyzed and optimized for hardware/software. VEINN's operations are instead continuous and learned (or pseudo-random) functions. In terms of quantum resistance, AES-256 against Grover effectively offers ~128-bit strength; VEINN's designers hope that their high-dimensional mapping can *further* slow down Grover by requiring the adversary to handle huge vector states or network inversion. However, unlike AES, VEINN has no formal security proof and no conventional design criteria (like differential/linear cryptanalysis bounds). Its security is heuristic – based on the assumption that a random-looking invertible mapping is hard to invert without the key.

- **Post-quantum block ciphers:** There are few "quantum-safe" block ciphers specifically. Some research (e.g. AES-QPP, Saturnin) tweaks AES-like designs to resist quantum circuits [12]. VEINN takes a more radical route by abandoning algebraic structure altogether. If effective, it would be a *symmetric primitive* that could itself resist quantum search by making the search space effectively enormous. But it remains experimental.

- **Public-key (asymmetric) PQC:** VEINN is symmetric-key only (shared secret), so it is not directly comparable to lattice or code-based KEMs used for key exchange. However, VEINN *could* be used in hybrid systems: for example, after establishing a key with CRYSTALS-Kyber (a KEM), one could encrypt data with VEINN-based symmetric encryption. In this sense, VEINN is complementary: it is an extra layer of encryption, not a full key-exchange protocol.

- **Entropy and Structure:** Standard PQC schemes (e.g. lattice, code) have clear algebraic structure, which has both benefits (efficient implementation, formal hardness assumptions) and drawbacks (potential algebraic attacks). VEINN's lack of structure is a double-edged sword: it might resist algebraic attacks because there is no neat mathematical description, but it also means traditional proofs don't apply. In the TRIP proposal, this is noted as an advantage: the "reduced algebraic structure" of a neural layer "thwarts known PQC attacks" [13]. If VEINN's mapping truly behaves randomly, it would hide partial leaks; but it also means we don't know how to bound its security.

Overall, VEINN is **experimental and unproven**. It is **not** currently a candidate in any formal PQC standardization. Its value to the cryptographic community is as a concept: a suggestion to use learned reversible functions as cryptographic mixing. It contrasts with mainstream practice (where neural nets, if used, are often confined to attacker/analysis roles, not core encryption).

Polymorphism and Extensions

A notable aspect of VEINN is its *flexibility*. The design is *polymorphic* in that one can change its form without altering the high-level mechanism. Possible extensions include:

- **Varying network architectures:** Instead of affine coupling layers, one could use other invertible transformations: additive coupling, invertible 1x1 convolutions (as in Glow), or even quantum-inspired reversible circuits if available. Each choice yields a different cipher structure. One could also change the depth or split scheme (channel vs spatial split), effectively generating a family of ciphers under the VEINN umbrella.

- **Hybrid composition:** VEINN can "wrap" existing ciphers. For instance, one could feed the output of an AES block cipher into a VEINN network layer (or vice versa). This hybrid would combine well-analyzed AES diffusion with VEINN's additional mixing. More generally, one could take any standard symmetric key algorithm and insert it as a layer in the INN pipeline, thereby obfuscating its algebraic properties. Conversely, VEINN could wrap a PQC scheme's output: e.g. after encrypting data with a lattice-based symmetric cipher (like applying a Keccak-XMSS or similar), the result could be passed through the VEINN network to provide extra entropy mixing.

- **Homomorphic or multi-party variants:** Though speculative, one could imagine training (or designing) the INN so that certain algebraic properties are preserved (for example, an additive homomorphism through the network) enabling limited computation on encrypted vectors. Research on invertible networks for privacy (like CryptoNets [14]) hints that neural structures can interact with homomorphic encryption; a VEINN-inspired design could similarly be explored. This is beyond the current prototype, but it shows extensibility.

- **Key schedule polymorphism:** Traditional ciphers often have simple or complex key schedules. VEINN's key schedule is essentially the way it derives network parameters from the secret. One could make this schedule itself dynamic or subject to neural evaluation, making the mapping evolve over time (like a polymorphic engine). This could, in theory, adapt to any plaintext pattern, further complicating analysis.

In short, VEINN can be seen as a **framework**: one could mix-and-match cryptographic components. Its core invertible-net layer could be just one module in a larger pipeline. This polymorphism means it is not a fixed algorithm but a style of design. As noted in the TRIP abstract, such a framework could act as a *"composable hardener"* on existing schemes [8]: an extra nonlinear layer that can be turned on or off.

Security Considerations and PQC Context

In the context of post-quantum cryptography (PQC), VEINN plays a non-traditional role. It is not a new mathematically-hard problem, but an *ad hoc* transformation intended to *increase* the difficulty of breaking encryption by brute force or by quantum means. Important points include:

- **Quantum resistance:** VEINN's goal is to resist quantum attacks by raising the complexity of any attempted inversion. In the best case, a quantum adversary faces a continuous high-dimensional search space with no known structure to exploit. Grover's algorithm on an n -bit keyspace would take $2^{n/2}$ quantum steps. If VEINN effectively uses, say, a 256-dimensional real key space (via network weights), the keyspace is uncountably large. However, in practice the key is a finite seed, so the true security relies on seed size. Still, the neural mapping could require adversaries to simulate large circuits or solve hard optimizations. There is no known quantum algorithm to directly invert a general invertible DNN more efficiently than searching the key parameters. This is a **heuristic** hope at this stage, not a proven guarantee.

- **Integration with PQC:** VEINN itself is symmetric. It could be used to encrypt messages under a shared secret that was exchanged via a post-quantum public-key protocol. In that sense, it "factors into PQC" as an optional layer: after using Kyber/Dilithium to establish a session key, the data could be encrypted with VEINN instead of (or in addition to) AES-GCM. It could also serve in a KEM-like role if one allows use of a pre-shared key to encrypt small data. However, unlike NIST's finalists, VEINN does not produce a public-key encryption or signature algorithm on its own.

- **Comparison to PQC performance:** Standard PQC schemes (like Kyber) operate on small blocks (e.g. key encapsulation of a few hundred bytes) and rely on structured noise and polynomial arithmetic. VEINN's main cost is running neural net transformations, which can be slower (many floating-point ops). It might be more suitable for high-performance scenarios (maybe on GPUs or specialized hardware) where matrix multiplies are fast. The parameter sweep report suggests performance is an open question.

- **Analysis and proofs:** Unlike AES (which has been studied for decades) or lattice-based schemes (with reductions), VEINN has no formal security analysis. Cryptanalysts would need to consider: can the invertible net be learned or approximated via chosen-plaintext attacks? (I.e. if an attacker can query the encryption oracle on inputs, could they train a network to predict outputs?) Neural nets are universal approximators, so with enough data one might recover F even without the key –

although the space is huge. Differential or linear cryptanalysis might be hard to apply because the mapping is not described by low-degree polynomials but by neural subnets.

Given these factors, VEINN must be seen as *highly experimental*. The cryptographic community would require extensive evaluation: how does decryption accuracy vary with network size? Can small changes in input cause wild output changes (avalanche-like behavior)? Does quantum amplitude amplification give any shortcut beyond classic brute force?

Conclusion

VEINN (Vector Encrypted Invertible Neural Network) offers a novel paradigm: use invertible neural nets as cryptographic mixing to potentially strengthen post-quantum security. Its high-dimensional, key-driven nonlinear mappings are unique and **non-trivial**, blending ideas from machine learning (normalizing flows) with symmetric-key design. As a proof-of-concept, it highlights a research direction rather than a finalized cipher. Key takeaways for the cryptographic community:

- **Abstract Hardness:** VEINN relies on the assumed intractability of inverting a complex reversible mapping without the key. This is different from classical hardness assumptions and needs new analytical tools.
- **Hybrid Potential:** It can serve as an **add-on** to existing protocols, possibly improving entropy without replacing core schemes.
- **Parameter Flexibility:** Its polymorphic nature suggests many variants to explore; one can adjust vector size, depth, or include other operations, making it extensible.
- **Open Challenges:** Formal security proofs, performance benchmarking, and quantum analysis are all open research problems.

In sum, VEINN is a thought-provoking experiment in cryptography. It may inspire further work on *neural-inspired* cryptographic layers and bridges between modern ML architectures and post-quantum defense strategies. However, it should be used with caution (and likely only in theoretical or layered contexts) until its security is better understood [10] [8].

References: See cited works and sources for background: Węgrzynek & Topa's neural-network hardening of symmetric ciphers [2]; NIST's selection of PQC algorithms [3]; and invertible-network cryptography concepts [6] [10].

[1] [7] [8] [13] GitHub - CaelumSculptoris/trip-pqc
https://github.com/CaelumSculptoris/trip-pqc

[2] [12] Postquantum symmetric cryptography inspired by neural networks
https://annals-csis.org/Volume_35/drp/pdf/9901.pdf

[3] NIST Releases First 3 Finalized Post-Quantum Encryption Standards | NIST
https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards

[4] [5] [6] [9] [10] cryptanalysis - Vector-Based Invertible Neural Networks for Experimental PQC - Cryptography Stack Exchange
https://crypto.stackexchange.com/questions/117670/vector-based-invertible-neural-networks-for-experimental-pqc

11 Neural cryptography - Wikipedia
https://en.wikipedia.org/wiki/Neural_cryptography

14 [PDF] CryptoNets: Applying Neural Networks to Encrypted Data with High ...
https://proceedings.mlr.press/v48/gilad-bachrach16.pdf