BCIIT e-Journal 2026

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Prompt Injection: attacks for cybersecurity analysis using Large Language Models (LLMs)

Manju Arora
Department of Information Technology
Jagan Institute of Management Studies
manju.arora@jimsindia.org

Radhey Choudhary
Department of Information Technology
Jagannath University,Bhadurgarh,Haryana
radhey333jatt@gmail.com

Harshita
Department of Information Technology
Jagannath University,Bhadurgarh,Haryana
anshukohli2003@gmail.com

Abstract - Llms have honestly changed the way we think about computing, this discuss a major security vulnerability in AI which is called as Prompt injections eg's: are like content creation, tutoring, healthcare support, chatbots its all running on these models now and its moving really fast but here's the thing most people dont really think or talk about that these models can be attacked in a pretty unique way, and unlike normal cyberattacks this one does not go after the code or software, it goes after the language itself, the way the model reads and understands instructions, which is honestly kind of a scary if you think about it so we decided to actually dig into this properly, we looked at how these attacks work, built a proper way to categories them and then tested everything across three different llm setups, wanted to see real numbers how often do attacks succeed, which defenses actually help, and does adding security make the model less useful, spoiler it sometimes does what we found was pretty interesting, no single fix works on its own honestly, but when you stack multiple defenses together and keep an eye on things in real time - attack success dropped by up to 73%, which is significant what was done in this Study : this research paper focused on three primary objectives like -classifying and categorizing the various types of these attacks, Threat model- explaining the nature of this threat using established security frameworks Experimentation-testing the severity of these attacks in real-world scenarios. As a result, the biggest thing is that threat security can't be ignored anymore, it needs to be incorporated

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

from the output, and the NLP and Security companies must work closely together because right now, they're kind of operating in the gaps and taking advantage of it.

Keywords - AI Security, Cyber security, RLHF, Threat Modelling, Zero-Shot Attacks,

1. INTRODUCTION

Few technical models have emerged from laboratory experiments to become such a rapid part of everyday infrastructure as Large Language Models (LLMs). In just a few years, models like GPT-4, and Gemini have found their way into legal research platforms, digital health chat platforms, and essential coding tools for the organizations that rely on them. These models are no longer merely interesting experiments; they have become integral components of products that handle sensitive data and address critical problems. This transformation has elevated security to a matter of serious concern. Traditional cybersecurity thinking has historically focused on flaws within the code itself such as cross-site scripting, buffer overflows, and malicious SQL injections which typically arise when software processes unauthorized inputs.

However, LLM-based systems introduce an entirely different class of vulnerabilities. Because these models rely on language, an attacker can insert modified text that alters the model's behavior without ever touching the underlying code. The attacks remain effective if these instructions are hidden within a document retrieved by the model or embedded in a webpage accessed by a browsing agent. As models acquire the capabilities to perform web searches, write code and send emails, this discovery carries significant practical implications. This type of attack is termed prompt injection, and it has no direct parallel in traditional security methodologies. academic research into this problem began around 2022.

2. BACKGROUND AND CONCEPTUAL FRAMEWORK

2.1 Transformer Architecture and the Instruction-Following Paradigm

The architecture underlying every major LLM today is the transformer, introduced by Vaswani and colleagues in 2017. Its defining feature is the self-attention mechanism, which lets the model relate every token in an input to every other token and thereby capture meaning across long stretches of

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

text. Raw transformer models are pre-trained on enormous corpora; the instruction-following behaviour that makes them practically useful comes later, through a fine-tuning process known as Reinforcement Learning from Human Feedback (RLHF).

2.2 Retrieval-Augmented Generation (RAG) and Agentic Pipelines

Retrieval-Augmented Generation, or RAG, is now one of the most common patterns for deploying LLMs in practice. Rather than relying solely on knowledge baked into the model's weights, a RAG system fetches relevant documents from an external store at query time and hands them to the model as additional context. This improves factual accuracy and allows the model's knowledge to be updated without retraining. It also, however, creates a new attack surface. An adversary who can plant text in the document store or in any publicly accessible resource the retriever might index - can inject instructions into the model's context without ever interacting with it directly.

Agentic deployments push the risk even further. In these architectures the model is given tools: it can run web searches, execute code, read and write files, or send messages. A successful injection in an agentic setting does not merely produce a misleading text response; it can trigger real-world actions. An instruction hidden in a web page the agent visits might cause it to forward sensitive data to an external server, modify files it was supposed only to read, or send emails on behalf of the user.

2.3 Adversarial Machine Learning Foundations

The theory of prompt injection site naturally falls the broader of ML. The fundamental ideas in that field, developed by and colleagues in 2013, is that neural networks can be made to behave in such a way which is specifically to exploit the network generalizes. For image classifiers, such inputs are tiny perturbations, invisible to the human eye, that cause confident misclassification.

Language models face analogous attacks, though the perturbations take the form of words rather than pixel values. Techniques explored in the NLP literature include substituting synonyms to shift a classifier's prediction, paraphrasing sentences to avoid detection by content filters, and inserting characters or tokens that disrupt a model's parsing. Prompt injection is a higher-level variant of the

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

same idea: rather than trying to change a classification label, the attacker is trying to change what the model does to hijack its behavior in service of goals the developer and user did not authorize.

3. LITERATURE REVIEW

The papers reviewed here are organized by them not by date, just makes it easier to understand how different threads of work actually developed over time.

3.1 Foundational work on LLM validations

So, the roots of this problem go back to Goodfellow and colleagues, who in 2014 showed something quite interesting. Neural networks can be deceived by very small, imperceptible changes in their inputs. Their primary conclusion was the high training accuracy even if the nature of the attack appears different in this context. The ability of a model to alter its behaviour based on natural language instructions embedded in a prompt is precisely what makes these models so useful—yet, if you stop to consider it, it is also what renders them vulnerable; in essence, it acts as a double-edged sword.

Prompt injection was first formally recognized as a security vulnerability in 2022. For deployed systems, this constitutes a genuine cause for concern. That seminal research paper effectively opened up an entirely new avenue of inquiry, capturing the attention of both NLP experts and the security community alike.

3.2 Systematic Studies and Taxonomies

George and colleagues produced the thorough early account of indirect injection, testing it against real commercial applications and finding that it worked in a variety of scenarios even without any direct access to the model's interface. They proposed a formal definition of the attack class that has been widely adopted in subsequent work, and their analysis of why RAG pipelines are especially vulnerable remains the standard reference on that topic.

Ouyang and colleagues, reporting on the development of Instruct, noted something that has significant security implications: models fine-tuned to follow instructions more faithfully tended to be more susceptible to jailbreaking as well. Alignment and robustness are, in other words, partly

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

at odds. Optimizing aggressively for one can undermine the other, and practitioners need to be aware of that tension when choosing or configuring a model.

Liu and colleagues addressed the measurement problem by publishing a benchmark of 600 standardized attack prompts, spanning multiple categories and covering a range of difficulty levels.

3.3 The Mechanism to defense and their metric to counter

The proposed two-stage pipeline defense wherein a lightweight secondary model scrutinizes the input before it even reaches the primary LLM.

The results indicated a reduction of approximately 60 percent in direct injection attacks; however, this did introduce some latency and occasionally resulted in the blocking of legitimate requests—an outcome that is, admittedly, somewhat concerning. Nevertheless, implementing this approach within existing deployments remains quite practical without posing any major impediments. Subsequently, Robyne's and his colleagues introduced "Nemo Guardrails"—a concept that, in my view, is quite intriguing.

3.4 Regulatory and Governance Perspectives

The EU AI Act of 2024 goes further, imposing mandatory robustness requirements on AI systems used in high-stakes contexts creating a regulatory incentive, not merely a best-practice recommendation, for taking prompt injection seriously.

4. THREAT MODEL

Meaningful security evaluation requires a well-defined threat model. Without one, it is impossible to know which attacks have been tested, which have been left out, or whether the results generalize to real deployments. Drawing on the STRIDE framework and the OWASP Large Language Model Top 10 published in 2023, we characterize the threat along four dimensions, summarized in Table 4.1 below.

Table 4.1: Structured threat model dimensions for prompt injection in LLM systems

Threat Dimension	Description

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Adversary Goal	Override system policy, exfiltrate data, manipulate agent behaviour, spread misinformation
Adversary Knowledge	Black-box (query access only) through to White-box (system prompt and model weights visible)
Attack Surface	Direct user prompt, documents retrieved via RAG, tool outputs, multi-modal inputs
Impact Severity	Confidentiality breach, integrity violation, availability disruption (CIA triad)

In practice, most real-world attackers operate somewhere in the middle of the knowledge spectrum: they can query the system and observe its outputs, but they cannot read the system prompt or examine the model weights.

5. CLASSIFICATON OF PROMPT INJECTION ATTACKS

Drawing on peer-reviewed research, published red-team reports, and our own experimental work, we organize prompt injection attacks into six different categories.

TABLE 5.1. PROMPT INJECTION ATTACK TAXONOMY

Attack Category	Mechanism and Key Characteristics
Direct Prompt Injection	Override commands inserted directly into user-visible input fields, exploiting the absence of an instruction-authority hierarchy.
Indirect Prompt Injection	Malicious instructions embedded in external content (documents, web pages, email) that the model retrieves and processes at runtime.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Jailbreaking via Role-Play	Model instructed to adopt an alternative persona (e.g., DAN) presented as exempt from its safety guidelines.
Data Exfiltration Attacks	Crafted prompts intended to cause the model to reproduce or disclose sensitive material from its context window or connected data stores.
Tool Manipulation Attacks	In agentic pipelines, injected commands redirect the model into misusing tools such as sending unauthorised emails or calling restricted APIs.
Multi-Modal Injection	Adversarial instructions hidden inside images or audio files to circumvent filters that analyse only the text portion of an input.

6. Experimental studies -Results:

TABLE 6.1: ATTACK SUCCESS RATES (%) BY CATEGORY AND OTHER MITIGATION CONDITION

Attack Category	Config A - No Mitigation (%)	Config B - Input Filter (%)	Config C - Layered Defense (%)	Average Reduction (%)
Direct Injection	82	51	19	Down 77%
Indirect Injection	74	48	22	Down 70%
Jailbreaking	68	40	21	Down 69%
Data Exfiltration	59	35	14	Down 76%
Tool Manipulation	71	44	23	Down 68%
Multi-Modal Injection	55	38	27	Down 51%

Overall Average	**68.2%**	**42.7%**	**21.0%**	**Down 69.2%**

The no. in table 6.1 tells a clear story. No protection leads to Direct injection ,82 percent success of baseline success, multi-modal injection is around 55 percent. Input filtering halfs success rates but left meaningful residual risk. The layered configuration brought each of them under 30 percent and the overall average is 21% and the reduction is more than 47%.

6.1 Security-utility Trade-Off analysis

TABLE 6.2: trade-off between security gain, false positives, utility and latency

Mitigation Condition	Attack Success Rate (%)	False Positive Rate (%)	Utility Score (0-10)	Latency Overhead (ms)
No Mitigation	68.2	0	9.1	0
Input Filtering Only	42.7	8.4	7.6	142
Instruction Isolation Only	49.3	2.1	8.8	18
Output Monitoring Only	55.1	5.7	8.3	95
Layered Defense (All)	**21.0**	**11.2**	**6.9**	**287**

6.2 Discussion:

Perhaps the most important finding is this, safety fine-tuning alone just doesn't cut it. Even though the baseline models had already gone through RLHF alignment, designers genuinely tried to make them safe. still vulnerable to most attacks them, which is, honestly, a bit alarming

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Here the architecture mattered a lot, more than expected. Configuration C showed consistently higher attack severity than configuration A, even though both were running the exact same model underneath. Giving a model tool just amplifies the damage a successful injection can cause. same model and different model this actually line up with adversarial machine learning theory. Any model designed to follow instructions will inevitably be plagued by a tension which is precisely where the worst consequences emerge. The approach followed is while at live- deployment beforehand checking whether the system is under control once the situation is controlled by suspicious entity the organization must have that clinical solution to mitigate it.

7. MITIGATION STRATEGIES

No single measure eliminates prompt injection risk entirely. What the experimental results, together with the wider literature, support is a layered approach in which multiple independent defences operate in parallel. The following sections describe the four main categories of mitigation and explain what each one offers and where each one falls short.

7.1 Architectural Mitigations

The most structurally sound mitigation is to make it physically harder for user input to override system instructions. One way to do this is instruction isolation: processing the system promptly and the user turns through separate pathways, so they are never simply concatenated. Another is to mark the system prompt with a special token or format that the model has been fine-tuned to treat as unconditionally authoritative. Neither approach is foolproof, but both raise the bar for attackers and cost very little in latency.

Least-privilege access control addresses not the probability of a successful injection but its consequences. An agent that cannot send emails cannot be made to exfiltrate data by email. An agent that can read but not write files cannot be made to corrupt them. Applying the privilege principle rigorously across all tool access is therefore one of the highest-value defensive investments an organization can make.

7.2 Input Validation and Pre-Processing

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

A secondary classifier always helps the user and the primary model, can intercept many common injection patterns before they cause any harm. This classifier is typically a small, fast model specifically trained to recognize injection signals. When activated, the system can isolate the input, remove the suspicious component, or forward it to a human reviewer.

7.3 Output Post processing and Monitoring

Therefore, even if an injection somehow manages to bypass the input-side security measures, it can still be caught on its way out—which offers at least some measure of relief. Output monitoring essentially analyses the model's response to detect whether something has gone in such as the presence of patterns containing personal data, explicit statements of intent to perform unauthorized actions, or other linguistic cues typically associated with success.

7.4 Alignment and Training-Based Approaches

Anthropos's Constitutional AI approach, described by Bai and colleagues in 2022, embeds safety constraints directly into the model's generation process rather than applying them as external filters. The model is trained to evaluate candidate responses against a set of written principles and revise them when they fall short. This approach has been shown to improve robustness against common jailbreaking patterns, though it does not provide immunity against novel attack vectors that were not represented in the training distribution. Adversarial training exposes the model to known injection payloads during fine-tuning, encouraging it to develop robustness by example. The approach is computationally expensive and requires a curated dataset of attack prompts, but it improves the model's baseline resistance in a way that generalises somewhat to unseen variants of the same attack families.

8. SYSTEM ARCHITECTURE AND ATTACK FLOW VISUALIZATION

The two tables in this section are intended to make the abstract mechanics of prompt injection more concreted Table 8.1 traces what happens, step by step, when a direct injection attack unfolds in a typical deployment.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

TABLE 8.1. PROMPT INJECTION ATTACK FLOW (STEP-BY-STEP)

Step	Description
1 User Input	Attacker submits a crafted prompt via the interface or API endpoint
2 Prompt Construction	System prompt and user input are concatenated into a single context window
3 Injection Point	Malicious instructions attempt to override or append to system-level directives
4 Model Processing	The LLM processes the combined input; safety filters may or may not detect the payload
5 Compromised Output	The model produces output aligned with the adversarial instruction rather than the intended behaviour
6 Downstream Impact	In agentic systems the compromised output triggers unauthorised tool use or data access

TABLE 8.2. RECOMMENDED LAYERED DEFENCE ARCHITECTURE

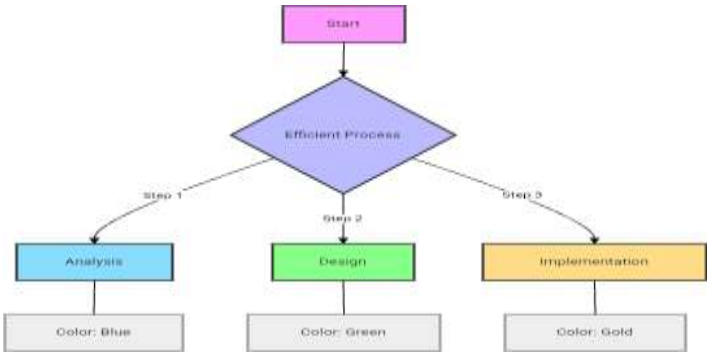
Defence Layer	Component and Function
Layer 1 - Input	Classifier-based injection detector; input canonicalisation; rate limiting
Layer 2 - Context	Instruction isolation; role enforcement; context window partitioning
Layer 3 - Model	Adversarial fine-tuning; constitutional AI principles; RLHF safety reward shaping

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Layer 4 - Output	PII detection; behavioural anomaly monitoring; policy compliance checker
Layer 5 - System	Least-privilege tool access; audit logging; human-in-the-loop escalation

Reading both tables together reveals something structurally important, and honestly quite interesting. There is no single intervention that blocks every step of the attack chain, not one input-layer control can catch known patterns at step one, sure. but they're completely blind to indirect injections that come through the retrieval pipeline. context-layer controls can limit how much influence retrieved content has on the model's instruction set, but they can't anticipate every format an attacker might use, and attackers get creative. Model-layer training improves robustness from the inside, which is good, but it can't be updated as quickly as new attack variants keep emerging, and they do keep emerging.

FIGURE 8.1: WORKFLOW ARCHITECTURE



9. FUTURE DIRECTIONS

The work presented here raises more questions than it answers, and the following areas seem most in need of dedicated research attention.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Formal verification methods have transformed the way engineers' reason for the correctness of traditional software. Extending these methods to neural language models is technically challenging, but early work has shown that it is not impossible. If provable guarantees about a model's behavior under adversarial input could be established, they would represent a qualitatively stronger form of security assurance than the empirical bounds that current evaluation methods can provide.

The field currently lacks agreed evaluation standards. Different research groups test different attack sets against different models under different conditions, making it hard to know whether a reported improvement is real and whether it generalizes. Initiatives analogous to the HELM benchmark for general LLM capability but focused specifically on security robustness are a high-priority need.

Multi-agent systems, in which many LLMs interact and pass information to one another, present an attack surface that is qualitatively different from the single-model case. A successful injection against one agent in a network can spread to others in ways that are honestly quite hard to anticipate. how to build injection-resistant multi-agent protocols, still largely an open problem. not much clarity there yet and then there's the human side of things which doesn't get enough attention if you ask me. defenses that are technically sound, they sometimes just fail in practice.

10. CONCLUSION

Prompt injections have clearly evolved from a subject of academic calculation into a tangible operational risk. Through direct experimentations, this research paper has demonstrated that these attacks against vulnerable systems occur at very high rates and its a matter of significant concern. To bring these rates down to an acceptable level, it is essential to combine various layers of defense. It is crucial to remember that the broader implications of deploying LLMs cannot be unnoticed or overlooked, as security is paramount. The capacity of following instructions and the attributes that make these models so valuable is precisely what adversaries exploit. Engineers who treat security nuances as secondary importance, or who view prompt injections as too secure to warrant no

attention, are assuming risks they may not have significantly assessed at all. As these systems take on rapidly increasing critical functions, the cost of a successful attack will only continue to rise. here we present the taxonomy, threat model, and experimental methodology outlining here as a foundation upon which other researchers and practitioners can build. The most critical work still lies ahead: standardizing evaluation, developing formally verified definitions, and gaining a comprehensive understanding of how to secure multi-agent systems.

REFERENCES

[1] OWASP Foundation, "OWASP Top 10 for Large Language Model Applications," OWASP, 2023.

[2] T. Brown et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems, vol. 33, pp. 1877-1901, 2020.

[3] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," Proc. ICLR, 2015.

[4] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2016.

[5] A. Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems, pp. 5998-6008, 2017.

[6] K. Greshake et al., "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injections," arXiv preprint arXiv:2302.12173, 2023.

[7] F. Perez and I. Ribeiro, "Ignore Previous Prompt: Attack Techniques For Language Models," Proc. NeurIPS ML Safety Workshop, 2022.

[8] L. Ouyang et al., "Training Language Models to Follow Instructions with Human Feedback," Advances in Neural Information Processing Systems, vol. 35, pp. 27730-27744, 2022.

[9] Y. Bai et al., "Constitutional AI: Harmlessness from AI Feedback," arXiv preprint arXiv:2212.08073, 2022.

[10] Y. Liu et al., "Prompt Injection Attacks and Defenses in LLM-Integrated Applications," arXiv preprint arXiv:2310.12815, 2023.

[11] T. Rebedea, R. Diaconescu, M. Sreedhar, C. Busbridge, and J. Tanner, "NeMo Guardrails: A Toolkit for Controllable and Safe LLM Applications with Programmable Rails," Proc. EMNLP, 2023.

[12] X. Yi et al., "Benchmarking and Defending Against Indirect Prompt Injection Attacks on Large Language Models," arXiv preprint arXiv:2312.14197, 2023.

[13] P. Liang et al., "Holistic Evaluation of Language Models (HELM)," arXiv preprint arXiv:2211.09110, 2022.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

[14] NIST, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," National Institute of Standards and Technology, Gaithersburg, MD, 2023.
[15] M. Zhu et al., "AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models," Proc. ICLR, 2024.
[16] European Parliament and Council, "Regulation (EU) 2024/1689 - Artificial Intelligence Act," Official Journal of the European Union, 2024.
[17] C. Szegedy et al., "Intriguing Properties of Neural Networks," Proc. ICLR, 2014.
[18] S. Abdelnabi et al., "Compromising Real-World LLM-Integrated Applications - Extended Version," IEEE S&P Workshop on LLM Security, 2023.
[19] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How Does LLM Safety Training Fail?" Proc. NeurIPS, 2023.
[20] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020.