# GOLLUM: Guiding cOnfiguration of firewaLL Through aUgmented Large Language Models

Roberto Lorusso[a], Antonio Maci[b] and Antonio Coscia[c]
*Cybersecurity Laboratory, BV TECH S.p.A., Milan, Italy*
{*roberto.lorusso, antonio.maci, antonio.coscia*}*@bvtech.com*

Keywords: Computer Networks, Conversational Agent, Cybersecurity, Firewall Configuration, Large Language Model, RAGAS, Retrieval Augmented Generation.

Abstract: Artificial intelligence (AI) tools offer significant potential in network security, particularly for addressing issues like firewall misconfiguration, which can lead to security flaws. Configuration support services can help prevent errors by providing clear general-purpose language instructions, thus minimizing the need for manual references. Large language models (LLMs) are AI-based agents that use deep neural networks to understand and generate human language. However, LLMs are generalists by construction and may lack the knowledge needed in specific fields, thereby requiring links to external sources to perform highly specialized tasks. To meet these needs, this paper proposes GOLLUM, a conversational agent designed to guide firewall configurations using augmented LLMs. GOLLUM integrates the pfSense firewall documentation via a retrieval augmented generation approach, providing an example of actual use. The generative models used in GOLLUM were selected based on their performance on the state-of-the-art NetConfEval and CyberMetric datasets. Additionally, to assess the effectiveness of the proposed application, an automated evaluation pipeline, involving RAGAS as test dataset generator and a panel of LLMs for judgment, was implemented. The experimental results indicate that GOLLUM, powered by LLama3-8B, provides accurate and faithful support in three out of four cases, while achieving $>80\%$ of answer correctness in configuration-related queries.

## 1 INTRODUCTION

Modern technologies that leverage artificial intelligence (AI) can positively contribute to the implementation of tactical cybersecurity services. Accordingly, several AI-based methods are being used to tackle the challenges posed by such a domain, including deep learning (DL) and natural language processing (NLP). Bridging them resulted in breakthrough advances in the performance achieved by computer-based agents in terms of their efficiency in handling challenging tasks related to human language. Currently, a clear example is given by the large language models (LLMs), representing the most widely distributed tools capable of understanding and generating general-purpose languages. The underlying design of such models is based on the use of transformers, i.e., artificial neural networks that implement the self-attention mechanism to extract linguistic content and to infer the relationship between words and sentences contained within. As it learns the patterns by which words are sequenced, the model can make predictions about how sentences should probably be structured (Min et al., 2023). In the cybersecurity domain LLMs can be mainly adopted for (Sarker, 2024): threat analysis, incident response, training and awareness, phishing detection, penetration testing, and implementation of conversational agents to provide real-time assistance to users. For the last use case, it has been observed that interaction with conversational agents can provide concrete support to humans in relation to the task at hand (Ross et al., 2023). Mitigating human error through assistance tools is crucial, especially when these errors occur in large and interconnected contexts, such as in the case of misconfigurations generated by network administrators on tactical devices, like firewalls (Alicea and Alsmadi, 2021). This represents a primary concern as it can result in serious performance and security problems, making the need for anomaly resolution and optimization algorithms that act on firewall security policies (Coscia et al., 2023; Coscia et al., 2025). In any case, it is paramount to avoid in advance the existence of

[a] https://orcid.org/0009-0007-8640-7109
[b] https://orcid.org/0000-0002-6526-554X
[c] https://orcid.org/0000-0002-7263-4999

Lorusso, R., Maci, A. and Coscia, A.
GOLLUM: Guiding cOnfiguration of firewaLL Through aUgmented Large Language Models.
DOI: 10.5220/0013221900003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 1*, pages 489-496
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

these abnormal setups, as they would hinder the device from being used in safety-critical contexts governed by strict standards (Anwar et al., 2021). In this setting, LLMs can be placed as powerful engines to assist operators with network configurations (Huang et al., 2024). Guiding the setup of specific firewall functionalities can be accomplished by employing the LLM as an agent that replies to specific questions posed by the operator, i.e., performing a question answering (QA) task (Ouyang et al., 2022). As a general rule, LLMs embody the knowledge acquired during training, thus limiting their reliability when employed in unknown contexts, potentially leading to a phenomenon called hallucination (Ji et al., 2023). This problem can be mitigated by expanding the model knowledge through fine-tuning, i.e., updating its weights using task-specific data. However, it requires more time and computational resources when scaled up with larger models. A viable alternative is represented by the so-called retrieval augmented generation (RAG) technique (Lewis et al., 2020), consisting of using dynamically retrieved external knowledge from custom documents in response to an input query. This approach mitigates hallucinations by providing more accurate models, which are also tolerant of fluctuations in context-specific information, ensuring consistency. In response to the challenge posed, leveraging the capabilities of the increasingly disruptive AI paradigm, this article proposes an application to assist network administrators in configuring firewall functionalities, namely, guiding configuration of firewall using augmented large language models (GOLLUM). To achieve this, the prior knowledge of small-sized LLMs was evaluated in the corresponding instruct version to assess their expertise in suggesting network configurations and answering cybersecurity questions, using the NetConfEval and CyberMetric datasets, respectively (Wang et al., 2024; Tihanyi et al., 2024). Through an ad hoc pipeline RAG, the most accurate and fastest models on the two datasets were equipped with external knowledge provided by the pfSense documentation. In summary, the main contributions of this paper are:

- The proposal of a conversational agent, namely GOLLUM, which can assist network administrators in firewall configurations.
- The implementation of an automated evaluation pipeline for the RAG chain that exploits LLMs in both the test case generation and the judgment phases.

## 2 LITERATURE REVIEW

Due to the radical diffusion of LLMs, recent studies have evaluated their deployment in emerging telecommunication technologies, such as 6G (Xu et al., 2024). In such a scenario, several collaborative LLMs, each with different scopes, are distributed among the network infrastructure to perform user-agent interactive tasks. In (Wu et al., 2024), a low-rank data-driven network context adaptation method was proposed to significantly minimize the fine-tuning effort of LLM employed in complex network-related tasks. This framework, called NetLLM, turned out to be useful in three specific networking use cases. Likewise, a study by (Chen et al., 2024) proposes the application of language models in the management of distributed models in cloud edge computing. This proposal is called NetGPT (according with the LLM leveraged) has the objective of releasing a collaborative framework for effective and adaptive network resource management and orchestration. The comprehensive investigations conducted by (Huang et al., 2024) discuss how LLMs impact and can enhance networking tasks, such as computer network diagnosis, design and configuration, and security. In (Ferrag et al., 2024), a Bidirectional Encoder Representations from Transformers (BERT)-based architecture is leveraged as a thread detector. It was fine-tuned on a Internet-of-things (IoT) data generated trough a combination of novel encoding and tokenization strategies. The AI-based agent presented in (Loevenich et al., 2024) is equipped with an AI-based chatbot that leveraged LLM and knowledge graph-based RAG technique to provide a human-machine interface capable of performing the QA task, according to the findings of the autonomous agent. In (Paduraru et al., 2024), an augmented LLama2 model provides a chatbot to support security analysts with the latest trends in information and network security. This was achieved by combining RAG and safeguarding with the LLM capabilities. According to the review we conducted, there appears to be a strong need to focus research activities on testing LLMs across various networking contexts. In addition, the same tools can greatly stimulate support activities, thereby fostering the achievement of a better cyber posture by users. To the best of our knowledge, no previous study has investigated the use of LLMs for assisting in the configuration of critical network security devices, such as firewalls. In such a scenario, a user could ask to an AI-based agent to support him in: (i) configuring network policies; (ii) indicating which are the steps to follow to configure a specific functionality according to the manual of a specific product (e.g., pfSense); (iii) understanding which are the attack vectors related to a line of defense (e.g., what are SQL injection (SQLi) attacks and how prevent them enabling web application firewall (WAF) proxy-based plugin like (Coscia et al., 2024)). Depending on the operative context and the proposed purpose, it is essential to develop safety-oriented agents, preferably through local LLMs, since these can be deployed in closed private scenarios.

## 3 GOLLUM DESIGN

GOLLUM consists of a RAG pipeline equipped with a parent document retriever that augments the context understanding capabilities of a language model. Regarding the language model adopted in GOLLUM a deeper discussion on the selection criterion is provided in section 4.1.2.

### 3.1 Knowledge Base

The knowledge base is the main source of information during retrieval. It complements the implicit knowledge encoded by the model parameters with structured or unstructured information, such as textual data or even images. To provide a real-world use case of GOLLUM we refer to the pfSense firewall. In particular, we retained the instruct-oriented text of interest based on its clarity, simplicity, and conciseness and deprived of any not-relevant reference to pure firewall topics. Moreover, each book content has been pre-processed so that only chapter content is retained, thereby removing prefaces, frontispieces, and any outlines. As a consequence of the overall content analysis and to avoid redundancies, the book (Zientara, 2018) was chosen as the final knowledge base in view of its substantial textual content and the limited presence of figures and tables. A descriptive analysis of topics covered in this book is presented in Table 1.

Table 1: Descriptive analysis of knowledge base.

| Topic | No. pages | No. tokens |
|---|---|---|
| Captive Portal | 35 | 49536 |
| Configuration | 38 | 50648 |
| DNS | 31 | 48465 |
| Firewall/NAT | 51 | 72184 |
| Installation | 40 | 62118 |
| Multiple WANs | 29 | 45267 |
| NTP/SNMP | 12 | 16578 |
| Routing and Bridging | 43 | 62523 |
| Traffic Shaping | 27 | 53073 |
| Troubleshooting | 74 | 101840 |
| VPN/IPsec | 49 | 68859 |

### 3.2 Document Chunker

Chunking documents is a standard practice that aims to improve the accuracy of information retrieval (IR) systems and ensures that the augmentation process does not saturate the length of the context window of LLMs. However, fine-grained chunks may lose important contextual information; therefore, it is important to balance the trade-off in chunk length. A common approach involves linking smaller fragments to the original full documents or larger chunks, with the goal of maximizing the accuracy of the retrieval process while preserving the broader context to be used in the generation process. According to this procedure, we employ a recursive character splitter with a chunk size of 256 for child nodes and 1700 for parent nodes with an overlap of 100 characters to preserve continuity between adjacent chunks. The length of the parent nodes was chosen to be close to the mean value of the lengths of documents. In addition, the chunk length was set so as to take advantage of as much as possible of the context window of the adopted language model, while avoiding saturating it with the number of chunks that can be retrieved.

### 3.3 Embedding Model

In a RAG pipeline, the embedding model plays a crucial role, as it transforms text into a searchable format that allows efficient and relevant information retrieval. Such an encoded structure is a vector (a high-dimensional numerical representation) that captures the semantic meaning of a sequence of text (e.g., a query), ensuring that similar concepts are close to each other in the vector space, even if they are phrased differently. The embedding model selected for GOLLUM was the so-called mxbai-embed-large-335M ($d = 1024$) as it stably appears in the top lightweight performers on the massive text embedding benchmark (MTEB) leaderboard (Muennighoff et al., 2023). In addition, as stated by Mixedbread, such an embedding model is appropriate for RAG-related use cases. As for user queries, the embedding model encodes the knowledge base (such a phase happens offline); thus, all external knowledge documents are transformed into vectors that are stored in a vector database for fast lookup.

### 3.4 Vector Store

Vector stores represent a fundamental component in RAG applications, as they are needed for storing, indexing and managing any type of data in the form of high-dimensional, dense numerical vectors, commonly produced by an embedding model. These vectors convey a semantic representation of the data, allowing for similarity-based retrieval through metrics such as cosine similarity or maximum marginal relevance. Embeddings are stored and retrieved using Chroma, an open-source, lightweight vector database with integrated quantization, compression, and the ability to dynamically adjust the database's size in response to changing needs.

## 3.5 Retriever

The vector stores can store multiple representations of the same documents. We employ a parent-document retrieval strategy that generates and indexes two distinct embeddings: one for the child chunks and another for the parent documents obtained at 3.2. Then, cosine similarity is computed between the embeddings of the input query and the child nodes to retrieve the most relevant matches; these are then used to retrieve the larger information segments from the parent nodes, which are finally used in the augmentation process. The number and size of the retrieved documents must be balanced according to the LLMs context window and available hardware resources. A larger context window increases the computational load. For our purposes, we retrieve the top four relevant parent chunks, each consisting of 1700 characters, to be fed into a context window of size 8192, i.e., the lower bound of maximum context window sizes supported by the employed models. Hence, we ensure that the context window is not saturated, thus leaving room for generation. As shown in Figure 1, embeddings of chunks related to the same topic exhibit spatial proximity.

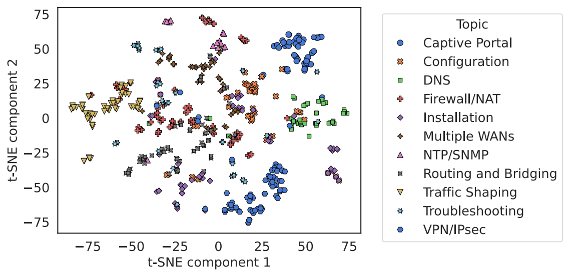

Figure 1: Parent embeddings per topic computed using the model outlined in section 3.3.

## 4 EXPERIMENTAL EVALUATION

### 4.1 Materials and Methods

#### 4.1.1 Datasets

To evaluate stand-alone LLM capabilities in both computer networks and cybersecurity domains, the following two datasets have been used: (i) NetConfEval[1] (Wang et al., 2024); (ii) CyberMetric[2](Tihanyi et al., 2024). Then, to assess the responses of the entire RAG pipeline against the golden answers, a custom test set comprising 330 question-golden answer pairs, 30 for each topic at Table 1, was constructed using the appropriate RAGAS (Es et al., 2024) utility. Due to the uneven distribution of content lengths across topics, as depicted in Table 1, it was necessary to develop an appropriately balanced test set. Taking into account possible noise, duplicates, and errors while generating the test set using the RAGAS utility, we started from a target number of 50 question-answer pairs for each topic. Then, we pre-processed the test set by removing duplicates, empty, and invalid golden answers. Finally, each question was manually inspected, reducing the number of QA pairs to 30 samples per topic. This was achieved by selecting the test samples based on the following criteria: (i) topic coherence; (ii) specificity, i.e., we retained the most specific and accurate QA pairs by discarding samples for which the related question does not require external knowledge to be answered. Given the same set of pfSense documentation used by the RAG chain, the LLMs involved in the generation of the synthetic test set were chosen to be different, but with a comparable size of LLMs in section 4.1.2, were: (i) Gemma-7B exploited as the generator; (ii) Gemma2-9B was used as a critic model to validate the generation process; (iii) Mxbai-embed-large-335M to produce the embeddings as in section 3.3. The generator-critic model pair is set so that the former is an older release than the latter, as recommended by the RAGAS utility.

#### 4.1.2 Large Language Models Evaluated

From the models evaluated in (Tihanyi et al., 2024), we selected the latest versions with available instruct models to translate requirements into formal specifications for reachability policies, emulating firewall traffic management. Consequently, the lightweight local LLMs assessed for the generation text component of GOLLUM are: (i) Llama3-8B and Llama3.1-8B; (ii) Mistral-7B and Neural-Chat-7B.

[1]https://github.com/NetConfEval/NetConfEval
[2]https://github.com/cybermetric/CyberMetric

### 4.1.3 Metrics

First, to evaluate the effectiveness of LLMs in dealing with the translation of network specifications, the accuracy is calculated by dividing the number of correctly translated requirements by the expected ones given a fixed batch size $b$ (number of specifications). In particular, each test case comprised $m = \frac{b_{MAX}}{b}$ samples, where, in our case $b_{MAX} = 10^2$ as $b \in [1, 5, 10, 20, 50, 100]$. On the other hand, because CyberMetric consists of multiple choices QA test, the average (since each experiment has been repeated five times to ensure good statistical significance, as done by the authors of the dataset in their experiments) LLM capability of selecting the correct answer among the four different options (accuracy). In addition, the medium inference time was recorded for both tasks because we were interested in models that were accurate and quick to respond in their practical employment. Then, according to (Adlakha et al., 2024), QA instruction-following and context-specialized models should be evaluated mainly along two aspects, which aim to point out the correctness of the provided answer and how the information conveyed by the agent is sourced from the external knowledge provided. The paradigm adopted for such an evaluation is the so-called LLM-as-a-judge (Zheng et al., 2023). However, to ensure a more impartial judgment, we formed a panel of judges, consisting of an ensemble of three local medium-sized LLMs not involved in any setup adopted so far, i.e.: Phi3-14B; Vicuna-13B; Qwen2.5-14B. We choose a trio of models, given the nature of the judgment requested to each LLM, that is, binary. This choice is necessary to implement a majority voting schema to infer definitive decisions. Therefore, at least two models can provide agreeable judgments. To realize evaluation purposes, the following metrics were considered:

- Answer correctness (AC), that is, an indicator of how well the generated answer compares to the ground truth. First, to compute such a measure, the factual correctness (FC) is derived as the F1 score calculated on the number of: (i) statements shared by the answer and the ground truth (true positives (TPs)); (ii) facts in the generated answer that do not belong to the expected answer (false positives (FPs)); (iii) statements found in the ground truth but not in the generated answer (false negatives (FNs)). These measures are derived according to the decisions inferred by the aforementioned majority voting scheme. It should be noted that TPs and FPs are mutually exclusive, which makes the decision binary. Second, the cosine similarity between the answer ($\mathbf{e}_A$) and the ground truth ($\mathbf{e}_{GT}$) encoded as embeddings is calculated and then averaged with the FC:

$$\mathrm{AC} = \frac{\frac{|\mathrm{TP}|}{|\mathrm{TP}| + \frac{1}{2} \times (|\mathrm{FP}| + |\mathrm{FN}|)} + \frac{\mathbf{e}_A \cdot \mathbf{e}_{GT}}{\|\mathbf{e}_A\| \|\mathbf{e}_{GT}\|}}{2} \tag{1}$$

- Faithfulness (FF), i.e, how consistent is the generated answer with respect to the given context. To realize this, the generated answer is initially split into $\mathrm{N_A}$ single statements. Then, the panel of LLMs judge determines how many statements ($\mathrm{n_{FF}}$) are effectively retrieved from the context:

$$\mathrm{FF} = \frac{\mathrm{n_{FF}}}{\mathrm{N_A}} \tag{2}$$

## 4.2 Results and Discussion

### 4.2.1 LLMs Knowledge of Computer Networks and Cybersecurity Analysis

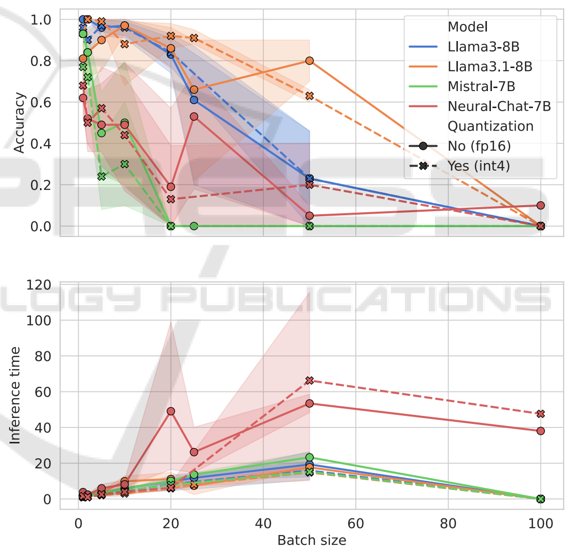

Figure 2: LLM accuracy and inference time (in seconds) achieved on NetConfEval per different weight quantization and batch size.

Figure 2 displays the average accuracy and inference time trends achieved by the evaluated LLMs for different $b$ values. For $b = b_{MAX} \rightarrow m = 1$, and this justifies the absence of the confidence interval in the correspondence of $b = 100$. However, and in accordance with the results obtained by the authors of NetConfEval, for $b = 100$, we noticed that the LLM outputs produced are always cut, which leads to performance degradation. Ranging the performance for each model, the following findings are derived:

- Llama3-8B achieved high accuracy at smaller batch sizes without quantization. As the batch size increases (beyond 20), there is a rapid drop in accuracy. The model quantization also follows a similar pattern but starts slightly lower. The inference time is relatively low at smaller batch sizes and increases gradually as the batch size increases. The fp16 weight precision requires slightly more time compared to int4 quantization, which indicates that the latter offers faster inference while retaining similar accuracy.

- Llama3.1-8B demonstrated relatively high accuracy at smaller batch sizes for both the fp16 and int4 configurations. This improvement is significant compared to Llama3-8B for $b = 50$. As previously observed, the accuracy decreased as the batch size increased, with a significant drop-off beyond batch size 50. The int4 version demonstrated better accuracy than fp16 (for $b \in [1,5,20]$) and produced the highest average accuracy among all compared models (higher than 60%) for $b = 50$. The inference time remained low and stable for all batch sizes, demonstrating that this language model was more efficient in handling larger batches than the other models. The int4 configuration remained slightly faster than the fp16.
- Except for small batch sizes, Mistral-7B achieved low accuracy scores among all compared models (from $b = 10$) regardless of whether model quantization is adopted or not. The model is completely insolvent even at b = 20. In addition, it requires a longer inference time than Llama models for both the fp16 and int4 setups.
- Neural-Chat-7B showed a sharp increase in inference time with increasing batch sizes, peaking at batch size 50 (around one minute) and stabilizing afterward. The int4 configuration demonstrated a faster inference time (except for $b \geq 50$) than the fp16 configuration, although the speed gains were accompanied by low fluctuating accuracy. This measure is the major drawback of this model because it is under 20% for $b \geq 20$.

As a general overview, the accuracy and inference time appear to decrease and increase, respectively, with increasing $b$. Considering the trade-off between accuracy and inference time, the models belonging to the Meta Llama3 family appear to be the best among the evaluated in providing network policy translations. This result is critical because it opens up the scenario of declining these models for the network analysis policy task, which is primary in firewall applications. Furthermore, adopting quantization does not lead to average performance degradation, ensuring faster inferences and, inevitably, lower GPU memory consumption.

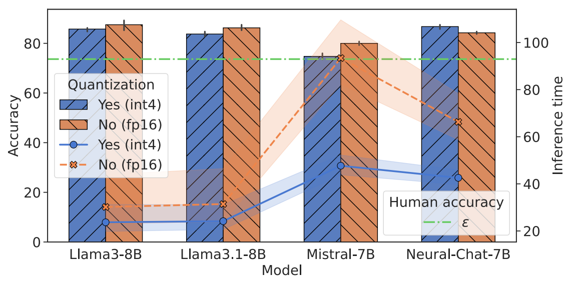

Figure 3: LLM accuracy and inference time (in seconds) achieved on CyberMetric per different weight quantization.

Figure 3 indicates that the accuracy achieved by Mistral-7B, which is close to $\varepsilon = 73.65$ (human accuracy) if quantization is enabled and the lowest among those achieved by all models compared in the case of the fp16 weight precision. Therefore, this model is the worst performer on the 80 questions of the CyberMetric dataset, examining also the inference time obtained, which is the longest compared to competitors per quantization adopted. The remaining three models outperformed human accuracy, and this result is notable for Neural-Chat quantized, which achieved the highest average accuracy score (with a negligible standard error) in these experiments. Despite this, the average inference time was longer than that achieved by the Llama models in both quantization settings. Finally, Llama3 and Llama3.1 achieve comparable performance in terms of accuracy and inference time (equal for this metric). Models without quantization require an average inference time of approximately 10 s, which indicates that they can answer ~480 questions in just a minute. On the other hand, this ability almost doubles that of adopting quantization while maintaining acceptable accuracy. Intersecting the results in both Figures 2 and 3, the Llama3 models appear to be more suitable in understanding contexts related to computer networks and security, producing accurate results in a very reactive manner.

#### 4.2.2 RAG-Based Analysis

Equipping GOLLUM with Llama3-8B or Llama3.1-8B results in a shift in AC and FF trends on the basis of how shown in Figure 4. To be specific:

- Llama3-8B achieves $\sim 76\%$ of AC with a standard error of approximately $\pm 0.02$. With regard to the alignment between the generated content and the context retrieved, this language model results in FF $\sim 75\%$.
- Llama3.1-8B yields $\sim 78\%$ of AC, i.e., the upper bound of the aforementioned model AC confidence interval, again with a standard deviation of $\pm 2\%$. In this case, despite the increase in accu-

racy compared to Llama3-8B, a drop in FF is evident, which appeared to be close to 70%. Similar to Llama3-8B, the standard deviation was wider than the AC.

According to the two above points, GOLLUM is more correct and faithful if Llama3-8B is used as the language model, i.e., a more balanced AC-FF trade-off is obtained compared to Llama3.1-8B usage. Establishing a balance between AC and FF can help ensure the response not only provides factually correct information and accurately represents the intended message, reasoning, or data. The achievement of 76% correctness implies that the model is accurate in three out of four answers, which is a promising starting point. This shows that the retrieval pipeline is generally effective; however, it could still be improved to capture more precise and relevant answers. A 75% faithfulness score means that, on average, one of four responses may include unfaithful or hallucinated content.

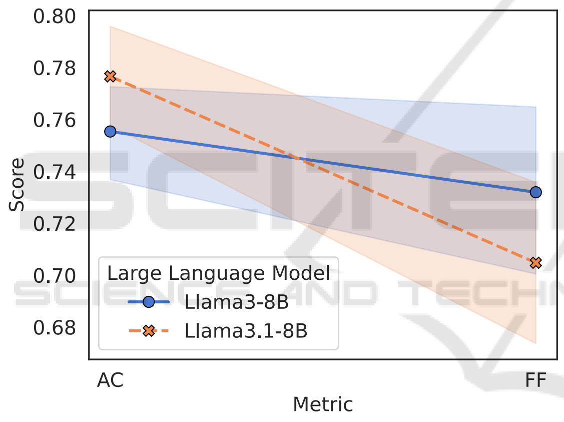

Figure 4: RAG metric scores per LLM leveraged by GOLLUM.

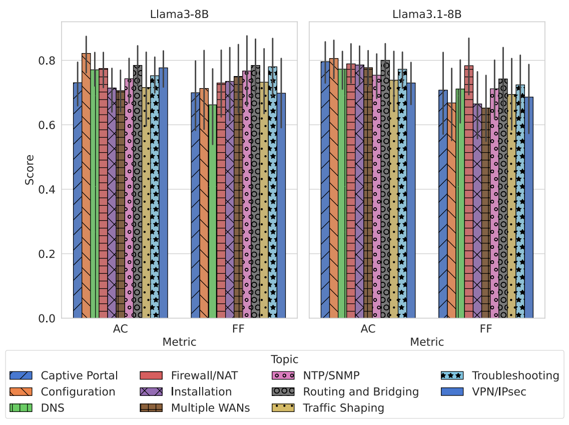

Figure 5: RAG metric scores per LLM achieved by GOLLUM on different topics.

Figure 5 depicts the average performance achieved by GOLLUM per topic. Specifically, Lllama3.1-8B outperforms Llama3-B in answer correctness on topics such as Captive Portal, Installation, and Multiple/WANs, while the opposite trend is observed for VPN/IPsec. Similarly, the FF achieved by GOLLUM equipped with Lllama3.1-8B is better for Firewall/NAT and DNS topics than the same score obtained by Llama3-8B, which provides more faithful answers in all the remaining topics. A remarkable result concerns the AC achieved on questions related to the configuration topic, which exceeds the 80% in both cases denoting how GOLLUM can provide accurate guidelines on configuration tasks. Similarly, very promising results are obtained for questions concerning routing and bridging topic, validating the efficiency in understanding network reachability requirements, as previously assessed with the NetConfEval benchmark, in a more specialized context (as the top FF score is exhibited). The consistency in model performance on various topics highlights the potential of GOLLUM to serve as a robust, context-sensitive support tool in network security.

## 5 CONCLUSION

Configuring firewalls is a challenging and error-prone task that can compromise network security. Given the complexity, especially in large-scale and dynamic networks, human operators can benefit from intelligent, context-aware tools for accurate firewall setup guidance. In this paper, we introduced GOLLUM, an AI-powered assistant designed to mitigate configuration challenges using a proper RAG pipeline. The proposed method exploits generative models that incorporate adequate prior knowledge of computer networks and cybersecurity. By combining LLMs with an extensive structured knowledge base, GOLLUM provides reliable and context-specific support to network administrators. Based on the experiments, GOLLUM equipped with Llama3-8B demonstrated more balanced performance, with considerable thoroughness in providing support on the topic of pfSense configurations. Future developments may include expanding the knowledge base to include additional network security resources, further optimizing the retrieval mechanism.

## ACKNOWLEDGEMENTS

This work was supported in part by the Fondo Europeo di Sviluppo Regionale Puglia Programma

Operativo Regionale (POR) Puglia 2014-2020-Axis I-Specific Objective 1a-Action 1.1 (Research and Development)-Project Title: CyberSecurity and Security Operation Center (SOC) Product Suite by BV TECH S.p.A., under Grant CUP/CIG B93G18000040007.

## REFERENCES

Adlakha, V., BehnamGhader, P., Lu, X. H., Meade, N., and Reddy, S. (2024). Evaluating correctness and faithfulness of instruction-following models for question answering. *Transactions of the Association for Computational Linguistics*, 12:681–699.

Alicea, M. and Alsmadi, I. (2021). Misconfiguration in firewalls and network access controls: Literature review. *Future Internet*, 13(11).

Anwar, R. W., Abdullah, T., and Pastore, F. (2021). Firewall best practices for securing smart healthcare environment: A review. *Applied Sciences*, 11(19).

Chen, Y., Li, R., Zhao, Z., Peng, C., Wu, J., Hossain, E., and Zhang, H. (2024). Netgpt: An ai-native network architecture for provisioning beyond personalized generative services. *IEEE Network*.

Coscia, A., Dentamaro, V., Galantucci, S., Maci, A., and Pirlo, G. (2023). An innovative two-stage algorithm to optimize firewall rule ordering. *Computers & Security*, 134:103423.

Coscia, A., Dentamaro, V., Galantucci, S., Maci, A., and Pirlo, G. (2024). Progesi: A proxy grammar to enhance web application firewall for sql injection prevention. *IEEE Access*, 12:107689–107703.

Coscia, A., Maci, A., and Tamma, N. (2025). Frog: A firewall rule order generator for faster packet filtering. *Computer Networks*, 257:110962.

Es, S., James, J., Espinosa Anke, L., and Schockaert, S. (2024). RAGAs: Automated evaluation of retrieval augmented generation. In *18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158.

Ferrag, M. A., Ndhlovu, M., Tihanyi, N., Cordeiro, L. C., et al. (2024). Revolutionizing cyber threat detection with large language models: A privacy-preserving bert-based lightweight model for iot/iiot devices. *IEEE Access*, 12:23733–23750.

Huang, Y., Du, H., Zhang, X., Niyato, D., Kang, J., Xiong, Z., Wang, S., and Huang, T. (2024). Large language models for networking: Applications, enabling techniques, and challenges. *IEEE Network*.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Loevenich, J. F., Adler, E., Mercier, R., Velazquez, A., and Lopes, R. R. F. (2024). Design of an autonomous cyber defence agent using hybrid ai models. In *2024 International Conference on Military Communication and Information Systems (ICMCIS)*, pages 1–10.

Min, B., Ross, H., Sulem, E., Veyseh, A. P. B., Nguyen, T. H., Sainz, O., Agirre, E., Heintz, I., and Roth, D. (2023). Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2).

Muennighoff, N., Tazi, N., Magne, L., and Reimers, N. (2023). MTEB: Massive text embedding benchmark. In *17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., et al. (2022). Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744.

Paduraru, C., Patilea, C., and Stefanescu, A. (2024). Cyberguardian: An interactive assistant for cybersecurity specialists using large language models. In *19th International Conference on Software Technologies - Volume 1: ICSOFT*, pages 442–449.

Ross, S. I., Martinez, F., Houde, S., Muller, M., and Weisz, J. D. (2023). The programmer's assistant: Conversational interaction with a large language model for software development. In *28th International Conference on Intelligent User Interfaces*, page 491–514.

Sarker, I. H. (2024). *Generative AI and Large Language Modeling in Cybersecurity*, pages 79–99. Springer Nature Switzerland.

Tihanyi, N., Ferrag, M. A., Jain, R., Bisztray, T., and Debbah, M. (2024). Cybermetric: A benchmark dataset based on retrieval-augmented generation for evaluating llms in cybersecurity knowledge. In *2024 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 296–302.

Wang, C., Scazzariello, M., Farshin, A., Ferlin, S., Kostić, D., and Chiesa, M. (2024). Netconfeval: Can llms facilitate network configuration? *Proceedings of the ACM Networking*, 2(CoNEXT2).

Wu, D., Wang, X., Qiao, Y., Wang, Z., Jiang, J., Cui, S., and Wang, F. (2024). Netllm: Adapting large language models for networking. In *ACM SIGCOMM 2024 Conference*, page 661–678.

Xu, M., Niyato, D., Kang, J., Xiong, Z., Mao, S., Han, Z., Kim, D. I., and Letaief, K. B. (2024). When large language model agents meet 6g networks: Perception, grounding, and alignment. *IEEE Wireless Communications*, pages 1–9.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., et al. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623.

Zientara, D. (2018). *Learn pfSense 2.4: Get up and running with Pfsense and all the core concepts to build firewall and routing solutions*. Packt Publishing.