

People's Democratic Republic of Algeria  
Ministry of Higher Education and Scientific Research  
Ferhat Abbas University of Setif 1



Université Ferhat Abbas Sétif 1

Faculty of Sciences  
Computer Science Department

## **DISSERTATION**

Presented in fulfillment of the requirements of obtaining the degree

**Master 2 in Computer Science**

Specialty: Data Engineering and Web Technologies

## **THEME**

---

Detecting SQL injections using BERT

---

*Presented by:*

ATHMANI RAMI

BOUHEZILA NASSIM

*Supervised by:*

Dr. BENZINE MEHDI

**2022/2023**

# Dedication

*To our parents,*

*To our grandparents,*

*To our brothers and sisters,*

*To our entire family,*

*To all our friends.*

*Athmani Rami,  
Bouhezila Nassim.*

# Abstract

Deep learning techniques have improved various domains by using their ability to learn complex patterns from large datasets. In this dissertation, we employed the power of Deep Learning, specifically BERT language model (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers), to resolve the issue of SQL injection attacks on web applications.

The goal of our study is to develop a Deep Learning model using BERT that can accurately identify SQL injections.

Based on the results, our model demonstrated excellent performance; it also indicated that BERT outperforms the compared machine learning models across different evaluation metrics. These results affirm the effectiveness of BERT in detecting SQL injection attacks, underscoring its superior performance in our study.

**Keywords:** Deep learning, Deep learning techniques, Deep Learning model, BERT language model, SQL injection attacks, Web applications, Machine learning models, Evaluation metrics.

# Résumé

Les techniques d'apprentissage profond ont amélioré divers domaines en utilisant leur capacité à apprendre des complexes patterns à partir de grands ensembles de données. Dans ce mémoire, nous avons utilisé la puissance de l'apprentissage profond, en particulier le modèle de langage BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers), pour résoudre le problème des attaques par injection SQL sur les applications web.

L'objectif de notre étude est de développer un modèle d'apprentissage profond en utilisant BERT qui identifie les injections SQL avec précision.

D'après les résultats, notre modèle a démontré d'excellentes performances ; ils ont également indiqué que BERT a surpassé les autres modèles d'apprentissage automatique comparés à travers différentes métriques d'évaluation. Ces résultats confirment l'efficacité de BERT dans la détection des attaques par injection SQL, confirmer sa performance supérieure dans notre étude.

**Mots-clés:** Apprentissage profond, Techniques d'apprentissage profond, Modèle d'apprentissage profond, Modèle de langage BERT, Attaques par injection SQL, Applications web, Modèles d'apprentissage automatique, Métriques d'évaluation.

## ملخص

حسنت تقنيات التعلم العميق مجالات مختلفة باستخدام قدرتها على تعلم الأنماط المعقدة من مجموعات البيانات الكبيرة. في هذه الأطروحة، استخدمنا قوة التعلم العميق، وتحديدًا نموذج اللغة "BERT" (Bidirectional Encoder Representations from Transformers)، لحل مشكلة هجمات حقن SQL على تطبيقات الويب.

الهدف من دراستنا هو تطوير نموذج التعلم العميق باستخدام BERT الذي يمكنه تحديد هجمات حقن SQL بدقة.

بناءً على النتائج المتحصل عليها، أظهر نموذجنا أداءً ممتازاً؛ كما أشار إلى أن BERT يتفوق في الأداء على نماذج التعلم الآلي التي تم المقارنة بها عبر مقاييس التقييم المختلفة. تؤكد هذه النتائج فعالية BERT في اكتشاف هجمات حقن SQL، مما يؤكد أدائها المتفوق في دراستنا.

**الكلمات الدالة:** التعلم العميق، تقنيات التعلم العميق، نموذج اللغة BERT، هجمات حقن SQL، تطبيقات الويب، نموذج التعلم العميق، نماذج التعلم الآلي، مقاييس التقييم.

# Table of contents

<b>General introduction .....</b>	<b>12</b>
<b>Chapter 1 SQL injection .....</b>	<b>14</b>
1.1 Introduction .....	14
1.2 Understanding how web applications work.....	15
1.3 How SQL injections work .....	16
1.3.1 Definition .....	16
1.4 Techniques of SQL injections .....	18
1.4.1 Tautologies .....	18
1.4.2 Error-based SQL injection .....	18
1.4.2.1 Mysql database errors.....	19
1.4.3 Blind SQL Injection .....	23
1.4.3.1 Content-based.....	23
1.4.3.2 Time-based .....	24
1.4.4 Union-based SQL injections .....	25
1.5 SQL Injection defense techniques .....	26
1.5.1 Escaping .....	27
1.5.2 Input validation .....	27
1.5.3 Parameterized queries .....	29
1.5.4 Web application firewalls WAF.....	30

1.5.5	Detection using machine learning .....	30
1.6	Conclusion .....	31
<b>Chapter 2 Deep Learning.....</b>		<b>32</b>
2.1	Introduction .....	32
2.2	Machine learning .....	32
2.2.1	Types of machine learning .....	32
2.2.1.1	Supervised learning.....	33
2.2.1.2	Unsupervised learning.....	33
2.2.1.3	Reinforcement.....	33
2.2.2	Machine learning algorithms.....	34
2.2.2.1	Linear Regression.....	34
2.2.2.2	Logistic Regression.....	35
2.2.2.3	Support vector machines.....	35
2.2.2.4	K-Means.....	36
2.2.3	Machine learning applications .....	37
2.3	Deep learning.....	37
2.3.1	Artificial neural networks.....	38
2.3.2	Activation functions .....	40
2.3.3	Deep learning architectures .....	42
2.3.3.1	Recurrent Neural Networks.....	43
2.3.3.2	Long Short-Term Memory Networks .....	44
2.3.3.3	Gated Recurrent Units.....	45

2.3.3.4	Transformers .....	46
2.3.4	Deep learning applications .....	49
2.4	Conclusion .....	49
<b>Chapter 3 Conception and Implementation.....</b>		<b>50</b>
3.1	Introduction .....	50
3.2	General conception of the solution.....	50
3.3	Chosen model: BERT.....	51
3.3.1	BERT architecture.....	52
3.3.2	BERT for Text Classification.....	53
3.3.3	Why BERT was chosen.....	54
3.3.4	Fine-tuning BERT for SQL Injection Detection: .....	54
3.4	Presentation of development tools.....	54
3.4.1	Programming language .....	54
3.4.2	Libraries .....	55
3.4.3	Development environment .....	56
3.5	Dataset .....	57
3.6	Code and Implementation.....	59
3.6.1	Split and Preprocess data for the BERT model.....	59
3.6.2	Build the BERT model.....	60
3.6.3	Fine-tuning the BERT model .....	60
3.6.4	Make predictions with the BERT model.....	61
3.7	Choice of hyperparameters .....	62



3.7.1	Preprocessing hyperparameters.....	62
3.7.2	Data splitting hyperparameters.....	62
3.7.3	Model training hyperparameters .....	62
3.8	Conclusion.....	63
<b>Chapter 4 Test and Evaluation.....</b>		<b>64</b>
4.1	Introduction .....	64
4.2	Confusion matrix .....	64
4.3	Evaluation metrics for assessing model performance .....	65
4.3.1	Accuracy.....	65
4.3.2	Precision.....	65
4.3.3	Recall.....	66
4.3.4	F1 Score.....	66
4.4	Model performance analysis.....	66
4.5	Evaluate the presence of overfitting .....	68
4.6	Comparative analysis with other approaches .....	69
4.7	Model Performance evaluation on new Data .....	70
4.8	Conclusion.....	71
<b>General Conclusion.....</b>		<b>72</b>
<b>References.....</b>		<b>74</b>

## List of figures

Figure 1.1 Three-tier architecture. ....	15
Figure 1.1.2 Example of a SQL injection attack [3]. ....	17
Figure 1.3 How Information flows during an SQL injection error [4]. ....	18
Figure 1.4 SQL injection vulnerability in PHP code. ....	20
Figure 1.5 handle query error with mysqli library in PHP.....	21
Figure 1.6 Character-escaping in PHP code example.....	27
Figure 1.7 Input validation of a String example in PHP code. ....	28
Figure 1.8 Input validation of an integer example in PHP code. ....	28
Figure 1.9 Parameterized queries using PDO in PHP code example.....	29
Figure 2.1 Graphical representation of linear regression. ....	34
Figure 2.2 Graphical representation of logistic regression. ....	35
Figure 2.3 Graphical representation of Support vector machines.....	36
Figure 2.4 Graphical representation of k means. ....	36
Figure 2.5 Typical biological-inspired neuron.....	38
Figure 2.6 Schematic representation of a neural network.....	39
Figure 2.7 Sigmoid activation function.....	41
Figure 2.8 ReLU activation function. ....	41
Figure 2.9 Tanh activation function. ....	42
Figure 2.10 Diagram of simple recurrent network.....	43
Figure 2.11 Long Short-term Memory Neural Network.....	44

Figure 2.12 Gated Recurrent Unit.....	45
Figure 2.13 Architecture of transformers [19]. .....	46
Figure 3.1 Sql injection Detection Tool conception and architecture.....	51
Figure 3.2 BERT model size.....	52
Figure 3.3 BERT model architecture. ....	53
Figure 3.4 Dataset query classes distribution.....	58
Figure 3.5 Split data into training and testing sets and preprocess data for BERT model. ....	59
Figure 3.6 Build BERT model Python code. ....	60
Figure 3.7 Fine-tuning BERT model Python code. ....	60
Figure 3.8 Make predictions with the trained model. ....	61
Figure 4.1 Training and validation loss .....	68

## List of tables

Table 1.1 Results of a SELECT query without UNION.....	25
Table 1.2 Result of user query after a UNION based SQL injection.....	26
Table 4.1 Confusion Matrix. ....	64
Table 4.2 Confusion Matrix (Classification Results).....	67
Table 4.3 Comparing the model performances using various metrics (%).....	69
Table 4.4 Confusion Matrix (Test Classification Results).....	70

## General Introduction

The rapid growth of web applications has revolutionized the way we interact and conduct various activities online. From e-commerce platforms and social networks to financial systems and government portals, web applications have become an integral part of our daily lives. However, with greater dependence on online applications comes an increased danger of cyber attacks with SQL injections being one of the most common and dangerous vulnerabilities.

To mitigate the growing threat of SQL injections, traditional approaches such as input validation and query parameterization have been widely adopted. While these methods provide some level of protection, they often struggle to keep pace with the evolving attack techniques employed by adversaries. Thus, there is a pressing need for more advanced and proactive defense mechanisms to detect and prevent SQL injection attacks.

In recent years, deep learning approaches have emerged as a promising solution in various domains, leveraging their ability to automatically learn complex patterns from large datasets. One such powerful deep learning model is BERT (Bidirectional Encoder Representations from Transformers), originally developed for natural language processing tasks. BERT has proven to be highly effective in capturing the semantic and contextual understanding of text, leading to remarkable performance in tasks such as text classification.

In this research, we propose using the power of BERT-based deep learning models to address the critical issue of SQL injection attacks. Our objective is to develop a reliable and efficient detection model capable of accurately identifying SQL injection attempts in real-time. By using BERT's contextual understanding and semantic representation capabilities, we aim to create a model that can effectively distinguish between normal and SQL malicious queries.

Our thesis is organized as follows:

In Chapter 1, we explore SQL injection attacks, their definitions, types and their detecting techniques, then we head on machine learning and Deep Learning, we present popular

algorithmic approaches in machine learning and explore deep learning architectures in Chapter 2. Chapter 3 is dedicated to the general conception of our work and the materials used, including the dataset and the type of deep learning architecture employed. Furthermore, we cover the preprocessing steps taken to ensure the accuracy and efficiency of our system. In Chapter 4, we discuss the test and evaluation of our model for detecting SQL injection attacks. We use a variety of evaluation metrics, including accuracy, precision, recall, and F1 score. We also compare the performance of our model to other machine learning algorithms and related works.