

Model Stealing with Wise Query Selection

Melih Coşğun
Bilkent University
Ankara, Turkey
22301344

Ömer Kaan Gürbüz
Bilkent University
Ankara, Turkey
22301425

Mert Gençtürk
Bilkent University
Ankara, Turkey
22003506

Mehmet Can Şakiroğlu
Bilkent University
Ankara, Turkey
22301343

ABSTRACT

Model stealing is an important field of data privacy and machine learning, where an attacker replicates the functionality of a target model by querying it and using the responses to train a clone model. This work aims to improve the efficiency of model stealing by reducing the number of queries required. Traditional model stealing approaches rely heavily on extensive querying, which can be costly and risk detection. We propose two novel heuristics to optimize query selection: one that encourages dissimilarity with previous queries to ensure diverse coverage of the input space, and another that targets similarity with queries that the clone model predicts poorly to improve areas where the clone model previously underperformed.

Our methodology involves maintaining a comprehensive table of past queries, their embeddings, and the outputs from both the target and clone models. By calculating an effectiveness score for each candidate query based on these heuristics, we select the optimal queries to send to the target model. This approach aims to minimize the number of queries while maintaining the effectiveness of the cloned model. Experimental results demonstrate that our heuristic-driven query selection reduces the query load compared to traditional methods, achieving comparable accuracy, loss, and F1 score metrics. This work highlights the potential of heuristic-based query optimization for increasing the stealth and efficiency of model stealing.

KEYWORDS

Model Stealing, Query Selection, Heuristic

1 INTRODUCTION

Model stealing is a type of adversarial attack where an attacker aims to replicate the functionality of a target machine learning model by querying it and using the responses to train a clone model. This type of attack presents significant risks, especially when proprietary models are involved, as it can lead to intellectual property theft and compromise of sensitive information. Hence, given the potential consequences, it is crucial to work on the topic, exploring both its limitations and capabilities and sharing the findings to raise awareness.

Traditionally, model stealing involves querying the target model with various inputs and using the outputs to train a substitute model iteratively. However, this traditional approach for model stealing typically involves extensive querying of the target model, which can be both time-consuming and costly. Extensive querying not

only increases the cost and time of the attack but also increases the risk of detection. Therefore, in our work, we address this challenge by proposing novel heuristics to reduce the number of queries necessary for effective model stealing. Our approach leverages precisely two heuristics to minimize the required queries by selecting the most informative ones. The heuristics are designed to ensure efficient exploration of the input space while simultaneously focusing on areas where the clone model previously exhibited high disagreement with the target model. By optimizing query selection with this approach, our work proposes a more efficient model stealing processing.

Moreover, our work focuses on black-box attacks, where the attacker has no access to the internal parameters or architecture of the target model but can interact with it via an API to observe its outputs. The target model used in this work is a hate speech detection model, which simplifies the problem without the loss of generality as it is a binary classification model but preserves the complexity of the nature of the problem to showcase the limitations and capabilities as it is based on natural language. By using auxiliary data, pretrained embedding models, and our heuristics, we aim to minimize the number of queries required to train an efficient and effective clone model.

2 RELATED WORK

The related work section explores key areas of model extraction and data privacy. **Model Extraction Attacks** reviews methods for replicating models through querying, with foundational work by Tramer et al. (2016) and advancements by Krishna et al. (2019). **Query Efficiency** focuses on reducing the number of queries needed for model stealing, highlighting strategies by Krishna et al. (2019) and Cheng et al. (2018). **Defense Mechanisms** discusses approaches to mitigate these attacks, including the PRADA system by Orekondy et al. (2019) and meta-learning-based defenses by Du et al. (2020).

2.1 Model Extraction Attacks

Model extraction attacks, also known as model stealing, aim to duplicate the functionality of a target model through iterative querying and reconstruction. Tramer et al. (2016) pioneered this concept by demonstrating attacks on logistic regression and decision trees, highlighting the risks posed by ML-as-a-service platforms like Amazon and BigML. Their attacks successfully extracted models with near-perfect fidelity, emphasizing the need for better protection mechanisms in deployed models [8].

Similarly, Krishna et al. (2019) and Wallace et al. (2020) extended these attacks to NLP models, specifically those based on BERT. They showed that even with limited prior knowledge and queries, attackers could effectively steal the model, raising concerns over the intellectual property and competitive advantages held by such models [5, 9].

Papernot et al. (2017) introduced practical black-box attacks that leveraged substitute models trained on synthetic datasets generated through iterative querying of the target model. This approach proved effective across different ML models hosted by platforms like MetaMind, Amazon, and Google, showcasing the broad applicability of model extraction techniques [6].

2.2 Defense Mechanisms

Various defense mechanisms have been proposed to mitigate these risks. The PRADA system, introduced by Orekondy et al. (2019), represents one of the first generic defenses against model extraction attacks. PRADA detects anomalous querying patterns indicative of extraction attempts, providing a layer of security that is agnostic to the specific model or training data. This method showed effectiveness across different types of models, marking a significant step towards robust defense strategies [4].

Another approach, proposed by Du et al. (2020), focuses on query-efficient meta-attacks. Their method leverages meta-learning to reduce the number of queries needed to perform successful attacks, thereby enhancing the stealth and efficiency of the attack. This highlights the ongoing arms race between attack strategies and defense mechanisms in the ML security landscape [3].

2.3 Query Efficiency

Existing model extraction techniques often rely on extensive querying of the target model, which can be both costly and easily detectable. This has led to research on query-efficient model stealing methods.

Krishna et al. (2019) show that attackers can extract high-quality models even when using randomly sampled word sequences, highlighting the vulnerability of BERT-based models to this type of attack. While the work focuses on nonsensical queries, it still underlines the importance of query efficiency in model stealing.

Cheng et al. [2] propose a query-efficient approach for hard-label black-box attacks by reformulating the attack as a continuous optimization problem, enabling the use of zeroth-order methods like RGF. This method proves effective in attacking both deep neural networks and non-differentiable models like GBDT with significantly fewer queries.

Our approach differs from the methods in [5] and [2] by focusing on a more targeted and efficient query selection strategy within the context of model stealing for NLP models. While [5] leverages nonsensical queries for effective extraction and [2] presents an optimization-based attack for enhancing query efficiency in a hard-label black-box setting, our work explores the potential of leveraging a comprehensive understanding of the target model's behavior (obtained through previous queries) to minimize the total number of queries required for successful extraction.

3 METHODOLOGY

Our methodology for model stealing is structured around two key heuristics designed to optimize query selection. These heuristics are based on maintaining and utilizing a comprehensive table consisting of past queries, their corresponding embeddings, and their outputs from both the target model and the clone model. The process of our method is as follows:

3.1 Table Initialization

We begin by initializing a table that stores each query we send to the target model. For each query, the table records the query itself, its embedding, the output from the target model (T_out), and the output from the clone model (C_out). This table is crucial for informing our query selection strategy.

A representative table, Table 1, can be seen below.

Table 1: Query Embedding and Outputs

Query	Embedding	T_out	C_out
"Q1: ..."	[73, 123, ..., 109]	0.75	0.98
"Q2: ..."	[132, 95, ..., 204]	0.60	0.23
"Q3: ..."	[85, 142, ..., 176]	0.85	0.45
"Q4: ..."	[55, 121, ..., 150]	0.65	0.65
...

3.2 Query Embeddings

Each query is transformed into an embedding using a Sentence-Transformer model. This embedding captures the semantic content of the query.

3.3 Heuristic Effectiveness Score

$$\operatorname{argmax}_{a \in A} (-\alpha * \max_{q \in Q} (\operatorname{sim}(q, a)) + \beta * \max_{q \in Q} (\operatorname{sim}(q, a) * (q_{C_out}) - q_{T_out})) \quad (1)$$

For each candidate query, we calculate the Formula 1 as the effectiveness score that balances two factors, and we select the candidate query a that maximizes the formula. The two factors of the formula are as follows:

3.3.1 Dissimilarity with Previous Queries.

$$-\alpha * \max_{q \in Q} (\operatorname{sim}(q, a)) \quad (2)$$

To ensure diverse coverage of the input space, the candidate query should be different from those previously sent. This encourages exploration of new areas in the input space, enabling better learning of the target model's decision boundaries. This is represented by the part in Formula 2 within the effectiveness score. (α is the hyperparameter that specifies the importance of this part of the score.)

3.3.2 Similarity with High-Disagreement Queries.

$$\beta * \max_{q \in Q} (\operatorname{sim}(q, a) * (q_{C_out}) - q_{T_out}) \quad (3)$$

The candidate query should be similar to previous queries where the clone model's output had a high disagreement with the target

model's output. This allows the clone model to focus on areas where it previously performed poorly, thereby improving its accuracy. This is represented by the part in Formula 3 within the effectiveness score. (β is the hyperparameter that specifies the importance of this part of the score.)

3.4 Query Selection & Clone Model Training

The candidate query with the highest score is selected and sent to the target model. The response is then used to update the clone model, and the new query and its results are added to the table. This iterative process continues until the clone model achieves a level of performance.

A simple figure that depicts the training process of the clone model, Figure 1, can be seen below.

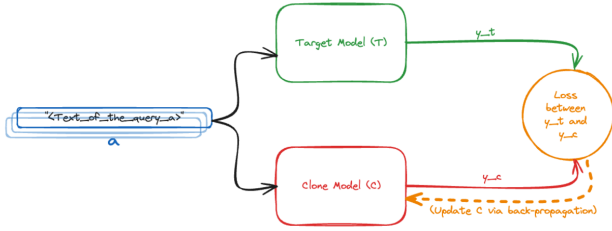


Figure 1: Training Process of the Clone Model

3.5 Evaluation

To assess the effectiveness of our approach, we conduct experiments comparing the heuristic-driven query selection against a baseline approach. The baseline involves querying without heuristics, either by using the entire dataset or by matching the number of queries used in the heuristic approach. We evaluate the number of queries used and the success metrics such as loss, accuracy, and F1 score.

By systematically applying these heuristics as explained above, we reduce the query load and enhance the efficiency of the model stealing process.

4 EXPERIMENTS

A series of comprehensive experiments were conducted to assess the effectiveness of our heuristic approach in relation to four distinct variables. We have tested a variety of settings for our heuristic score function. We have employed different neural network model architectures for our clone model, and we have explored different alpha and beta values for our score function. Finally, we have tested our approach on two distinct hate speech datasets.

It is crucial to emphasise that the selected datasets and model architecture are entirely distinct from the target model's architecture and training set, given that the target model is a black box and therefore unknowable.

We have conducted our experiments with learning rate of 0.0001, batch size of 32 and Adam optimizer.

4.1 Experiments with different neural network models

At the initiation of the experiments, the we defined the clone model as a one-layer GRU network with a single hidden linear layer. This architecture takes the input embeddings of the Glove [7] network outputs.

Following training on the data, it was observed that this architecture may not be a suitable fit for the problem. Consequently, an updated architecture comprising a Transformer based four-layer fully connected neural network with dropouts was implemented. The most significant difference between the updated and earlier architectures was that the latter takes the output embeddings of the sentence transformer of the input text as input. Since we used the same sentence transformer that we used in similarity calculation, the incorporation of this input transformation has resulted in a greater correlation between our architecture and the similarity calculation employed in our score function. The architecture of the second neural network can be seen in Figure 2.

4.2 Experiments with score formula

$$a = \arg \max_{a \in A} \sum_{q \in Q} (-\alpha \cdot \text{sim}(q, a) + \beta \cdot \text{sim}(q, a) \cdot (q_{C_{out}} - q_{T_{out}})) \quad (4)$$

We first tried our original formula (4), where we compute mean accross similarities for previous samples. Then, after some experiment we saw that formula has flaws. Especially, for samples that outlies the general distribution of data, our formula was performed relatively worse by only selecting those outlier samplees over and over again. In these cases we saw that some samples are too dissimilar to the all dataset such that no matter what is the α coefficient, they are always bee selected as next queries. Therefore, we solved this issue by selecting the maximum value of the similarity and continued our experiments with the updated version of our formula (5).

$$\arg \max_{a \in A} (-\alpha * \max_{q \in Q} (\text{sim}(q, a)) + \beta * \max_{q \in Q} (\text{sim}(q, a) * (q_{C_{out}} - q_{T_{out}}))) \quad (5)$$

4.3 Experiments with different alpha and beta values

After finalizing our query selection formula, we experimented with different α and β values to maximize the our clone models performance. To conduct experiments, we decreased α in every iteration of our query selection loop. Our aim was to favor samples that are dissimilar to previous queries at the beginning of selection to explore more diverse space of queries. Then, as more queries selected, rather than favoring dissimilar samples, we gradually favor samples that clone model and target model disagrees by increasing β value. In all below experiments, we regarded β as $1 - \alpha$. Table 2 shows some of the values we have experimented with.

4.4 Experiments with different datasets

In order to obtain more robust results, two different datasets of hate speech were subjected to experimentation. Both datasets comprised

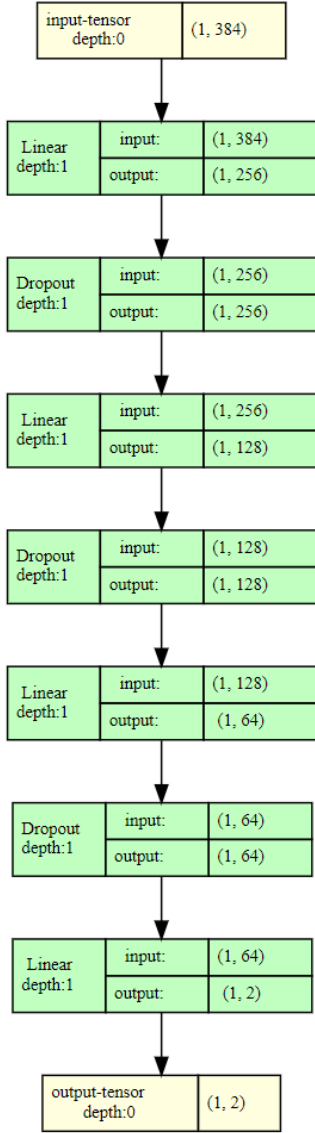


Figure 2: Architecture of our clone model. 4-layer fully connected neural network that takes the fixed size (1, 384) embeddings of the given text as input.

hate speech texts that had been collected from Twitter. The datasets were obtained from Kaggle [1]. The initial dataset, entitled *Hate Speech and Offensive Language Dataset*, encompasses 25,300 text data points. Given that the target model is providing the labels, there was no need to utilise the labels from the dataset. The second dataset, *Twitter Hate Speech*, contains 32,000 instances of hate speech. The experiments were conducted on these datasets using the final version of the methodology, and the results were shared in the Evaluation section of this report.

Table 2: Values of Initial and Final Alpha for different experiments

Experiment No.	Initial Alpha	Final Alpha
1	0.8	0.5
2	0.7	0.5
3	0.7	0.4
4	0.6	0.4
5	0.6	0.3

5 EVALUATION

To evaluate our query selection approach we started sending queries to the target model by our formula. Based on clone model’s output and target model’s output, we initialized a table containing both outputs, selected sample’s embedding and sample itself. After specified the number of iteration, we stopped query selection. Depending on parameters, we selected between 2000 to 14000 queries out of the auxiliary dataset containing 20,000 samples. Then, after we initialized our query table with our formula, we started model stealing. We used early stopping with early stopping patience of 5 to terminate the training.

To be able to compare the performance of our heuristic approach, we have trained 3 models per setting: one clone model is trained on selected queries (our heuristic approach), one clone model is trained on random samples with limited query count as our query selection table, and one clone model is trained on whole auxiliary dataset using target model’s outputs as labels. It is important to note that, we used the target model as a labeler on our test data, so that calculated scores indicates the similarity between our clone model and the target model.

This evaluation approach enables a comparison between our heuristic approach and the random query approach. In all cases, our heuristic approach yielded superior results when N queries were sent. Additionally, the scores were comparable to those obtained by querying the entire dataset. It is crucial to highlight that, given the associated costs of each query, reducing the number of queries is a key objective. It is to be expected that the results will improve with an increased number of queries, as this increases the size of the training data set for the clone model.

5.1 Results of different alpha and beta values

Table 3 shows all of the experiment results conducted with different α values. Method heuristic refers to model stealing conducted on selected queries. Remark that these results do not belong to query selection itself, rather, they are conducted after query selection with given parameters. Method random refers to model stealing conducted with selected number of queries chosen randomly without heuristic approach. Normal refers to model stealing on whole auxiliary dataset. Rest of the plots can be found in A.

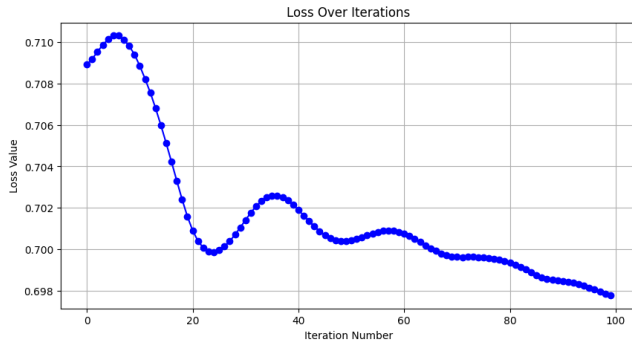
As illustrated in Table 3, the heuristic method consistently outperforms the random selection method across all pairs of initial and final α values in terms of both validation loss and test accuracy. This outcome validates the effectiveness of our heuristic approach in model training. Notably, the heuristic method achieves competitive accuracy with significantly fewer queries—ranging from 50% to

Table 3: Comparison of different methods for various initial and final α values

Initial α	Final α	Method	Val Loss	Test Acc	Queries
0.8	0.5	Heuristic	0.58510	0.7991	8267
0.8	0.5	Random	0.58609	0.7966	8267
0.7	0.5	Heuristic	0.58455	0.7962	7950
0.7	0.5	Random	0.58706	0.7922	7950
0.7	0.4	Heuristic	0.58356	0.7987	4994
0.7	0.4	Random	0.59048	0.7898	4994
0.6	0.4	Heuristic	0.58835	0.7858	2533
0.6	0.4	Random	0.59493	0.7797	2533
0.6	0.3	Heuristic	0.59155	0.7809	1746
0.6	0.3	Random	0.59157	0.7781	1746
-	-	Normal	0.57889	0.8031	19826

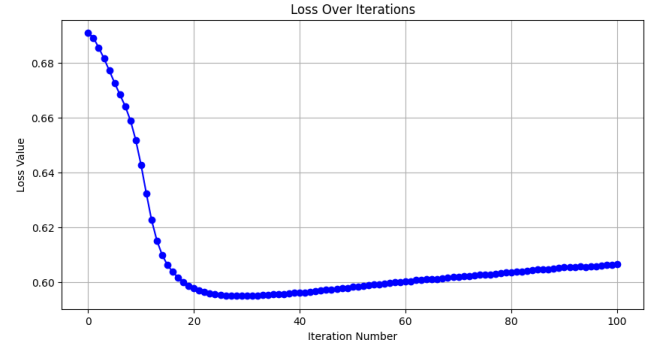
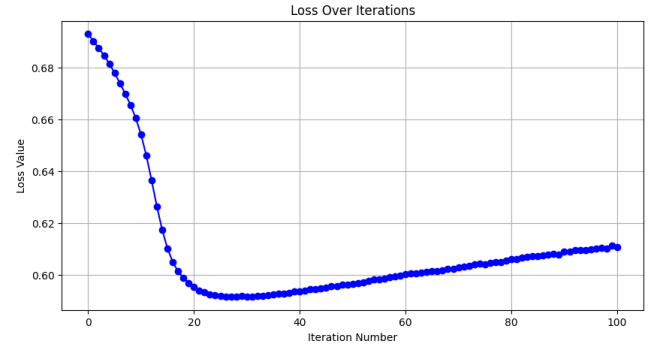
as little as 10% of the total auxiliary dataset—compared to the full dataset method.

Furthermore, the results demonstrate that while training the clone model on the entire auxiliary dataset with the target model’s outputs as labels generally leads to the highest performance, our heuristic approach manages to approximate these benchmarks with far fewer queries. This efficiency in query selection underscores the practical utility of the heuristic method, especially in scenarios where query costs are a concern.

**Figure 3: Query Selection Loss, $\alpha = 0.6$ to 0.4**

5.2 Results of different datasets

The experiments were conducted on two different datasets, as detailed in the Experiments section. In both datasets, query selection using a heuristic approach yielded superior results to random query selection. Additionally, the outcomes of querying the auxiliary dataset were presented. Nevertheless, the primary objective was to reduce the number of queries, given that querying the target model incurs a cost. The results of our experiments in both datasets are shown in Table 4. This table presents the results of the test sets that were initially partitioned from the data at the outside of the training process. It should be noted that the test sets were not presented to our models throughout the training period. The labels of the test sets were obtained from the target model. Figures 6 and

**Figure 4: Model Stealing Loss, with selected queries, $\alpha = 0.6$ to 0.4****Figure 5: Model Stealing Loss, with random queries, $\alpha = 0.6$ to 0.4**

7 illustrate the training loss graphics on the validation data for both dataset experiments. It can be observed that our query selection approach with a heuristic component is converging at a slightly faster rate than the random query selection approach.

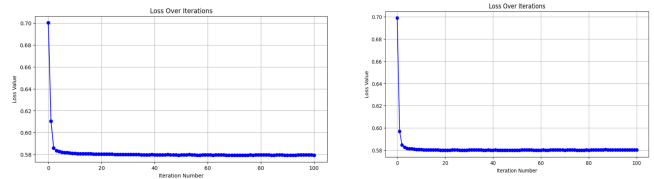
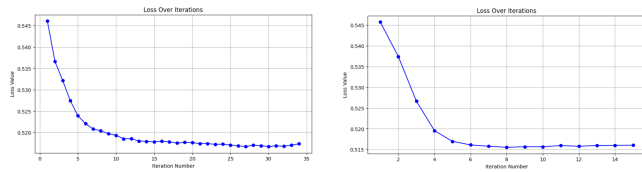
**(a) Training validation loss plot for heuristic query selection on 'Hate Speech and Offensive Language' dataset.****(b) Training validation loss plot for random query selection on 'Hate Speech and Offensive Language' dataset.****Figure 6: Comparison of training validation loss plots for heuristic and random query selection on 'Hate Speech and Offensive Language' dataset.**

Table 4: Experiment results of 'Hate Speech and Offensive Language' and 'Twitter Hate Speech' Datasets. Results are obtained on test data based on target model labels. Table on the top is results for 'Hate Speech and Offensive Language' dataset experiments while table on the bottom is results for 'Twitter Hate Speech' dataset experiment.

Query Selection	Number of Queries	Accuracy	F1
Heuristic	13267	0.8072	0.7246
Random	13267	0.8051	0.7183
All Data	19826	0.8047	0.7215

Query Selection	Number of Queries	Accuracy	F1
Heuristic	7673	0.8642	0.9205
Random	7673	0.8614	0.9197
All Data	25569	0.8692	0.9240



(a) Training validation loss plot for heuristic query selection on 'Twitter Hate Speech' dataset.

(b) Training validation loss plot for random query selection on 'Twitter Hate Speech' dataset.

Figure 7: Comparison of training validation loss plots for heuristic and random query selection on 'Twitter Hate Speech' dataset.

6 LIMITATIONS

The evaluation results demonstrate the efficacy of our heuristic query selection approach. As illustrated in Figure 7, training converges more rapidly with queries selected by the heuristic approach. However, the observed difference in accuracy is not as significant as expected. This could be due to several factors, including the capacity of our model architecture. As illustrated in Table 4, there is minimal distinction between training that employs a constrained query set and training that utilizes all auxiliary data. This suggests that our model may have reached its limits and is unable to learn further, independently of the quantity of data. This issue could be addressed by optimizing the model architecture and hyperparameters, such as the learning rate and dropout rate.

7 CONCLUSION

In this work, we addressed the challenge of model stealing by aiming to reduce the number of queries needed to clone a target model effectively. Traditional methods are query-intensive and can be costly and detectable. We introduced two novel heuristics for optimizing query selection: one encouraging dissimilarity with previous

queries and another targeting similarity with high-disagreement queries.

Our approach utilized a table tracking previous queries, their embeddings, and the outputs from both the target and clone models. By calculating effectiveness scores for candidate queries, we strategically selected the most informative ones. This method allowed us to efficiently explore the input space and focus on areas needing improvement.

Experimental results showed that our heuristic-driven query selection achieved a clone model with fewer queries compared to traditional methods. This reduced cost, time, and the risk of detection.

All in all, our findings demonstrate the potential of heuristic-based query optimization in making model stealing more efficient.

REFERENCES

- [1] [n. d.]. Kaggle. <https://www.kaggle.com>.
- [2] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. 2018. Query-Efficient Hard-label Black-box Attack: An Optimization-based Approach. arXiv:1807.04457 [cs.LG]
- [3] Jiawei Du, Hu Zhang, Joey Tianyi Zhou, Yi Yang, and Jiashi Feng. 2020. Query-efficient Meta Attack to Deep Neural Networks. arXiv:1906.02398 [cs.CV]
- [4] Mika Juuti, Sebastian Szyller, Alexey Dmitrenko, Samuel Marchal, and N. Asokan. 2018. PRADA: Protecting Against DNN Model Stealing Attacks. *2019 IEEE European Symposium on Security and Privacy (EuroS&P)* (2018), 512–527. <https://api.semanticscholar.org/CorpusID:21670607>
- [5] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. 2020. Thieves on Sesame Street! Model Extraction of BERT-based APIs. arXiv:1910.12366 [cs.CL]
- [6] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks against Machine Learning. arXiv:1602.02697 [cs.CR]
- [7] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1532–1543. <https://doi.org/10.3115/v1/D14-1162>
- [8] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. 2016. Stealing Machine Learning Models via Prediction APIs. arXiv:1609.02943 [cs.CR]
- [9] Eric Wallace, Mitchell Stern, and Dawn Song. 2021. Imitation Attacks and Defenses for Black-box Machine Translation Systems. arXiv:2004.15015 [cs.CL]

A LOSS PLOTS FOR α EXPERIMENTS

A.1 $\alpha = 0.6$ to 0.3

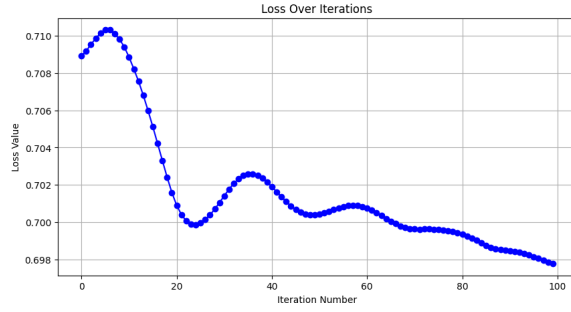


Figure 8: Query Selection Loss, $\alpha = 0.6$ to 0.3

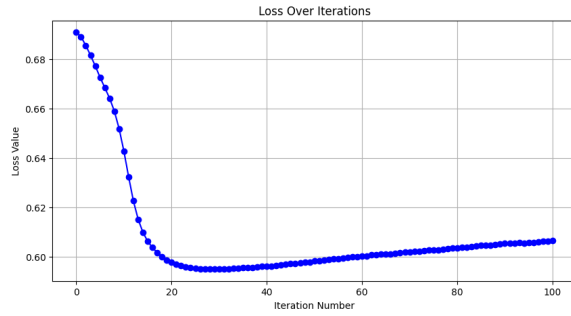


Figure 9: Model Stealing Loss, with selected queries, $\alpha = 0.6$ to 0.3

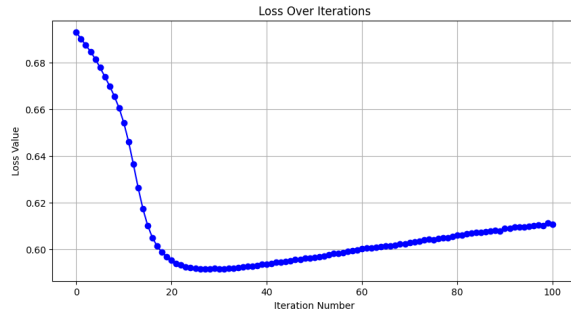


Figure 10: Model Stealing Loss, with random queries, $\alpha = 0.6$ to 0.3

A.2 $\alpha = 0.6$ to 0.4

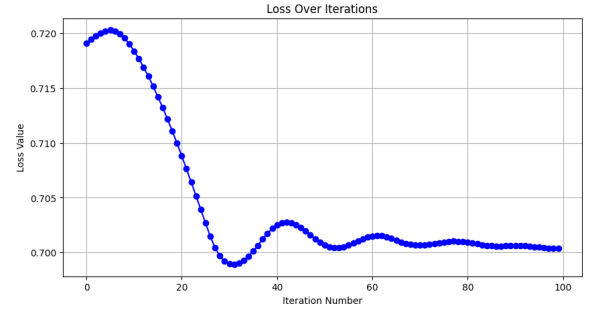


Figure 11: Query Selection Loss, $\alpha = 0.6$ to 0.4

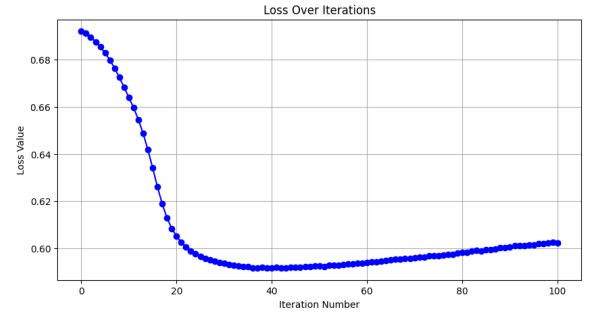


Figure 12: Model Stealing Loss, with selected queries, $\alpha = 0.6$ to 0.4

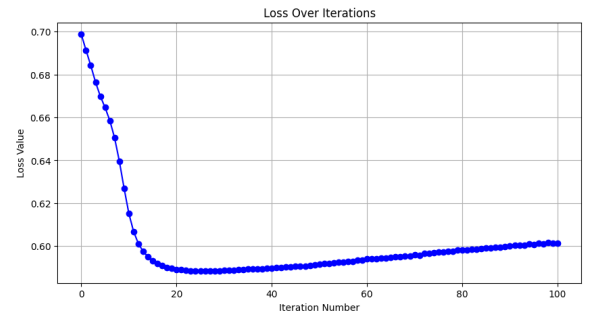
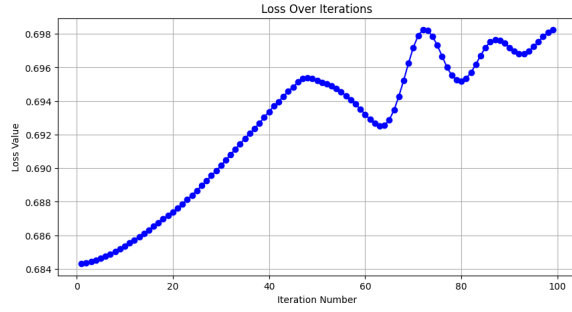
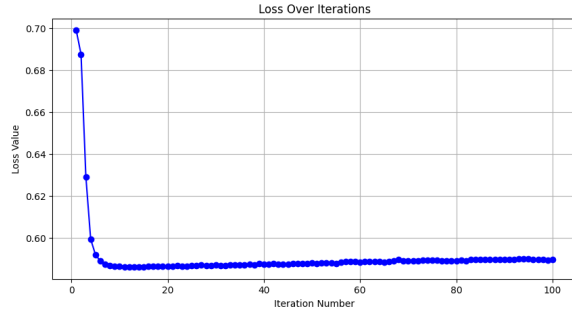
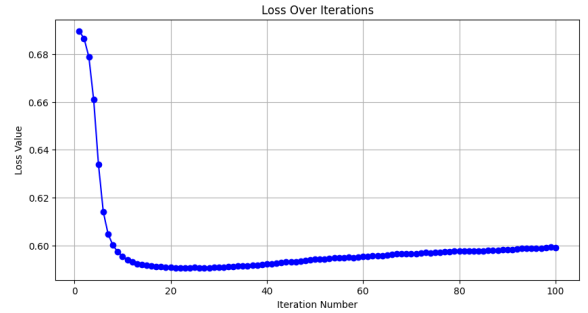
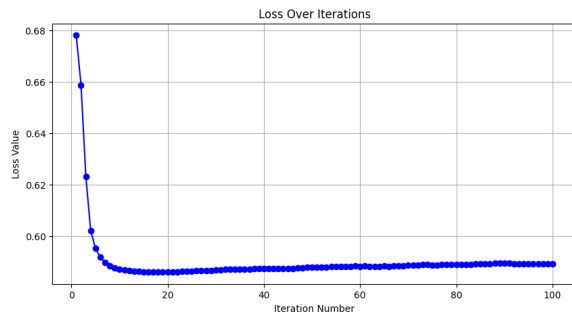
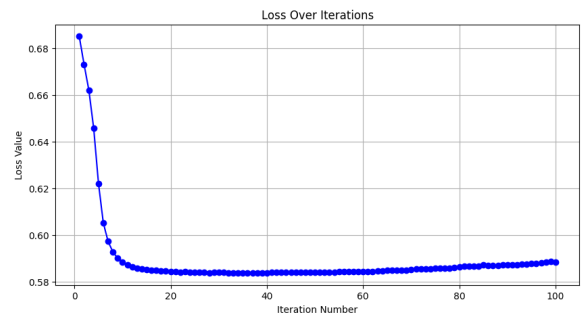


Figure 13: Model Stealing Loss, with random queries, $\alpha = 0.6$ to 0.4

A.3 $\alpha = 0.8$ to 0.5**Figure 14: Query Selection Loss, $\alpha = 0.8$ to 0.5****A.4 $\alpha = 0.7$ to 0.4****Figure 17: Query Selection Loss, $\alpha = 0.7$ to 0.4****Figure 15: Model Stealing Loss, with selected queries, $\alpha = 0.8$ to 0.5****Figure 18: Model Stealing Loss, with selected queries, $\alpha = 0.7$ to 0.4****Figure 16: Model Stealing Loss, with random queries, $\alpha = 0.8$ to 0.5****Figure 19: Model Stealing Loss, with random queries, $\alpha = 0.7$ to 0.4**

A.5 $\alpha = 0.7$ to 0.5

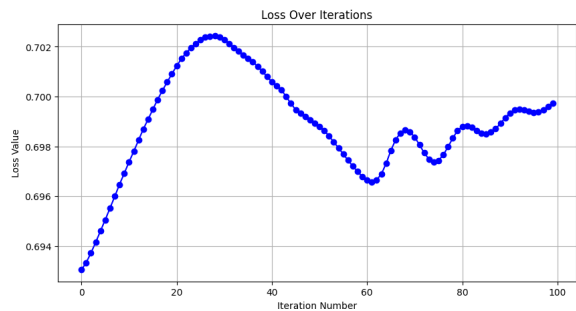


Figure 20: Query Selection Loss, $\alpha = 0.7$ to 0.5

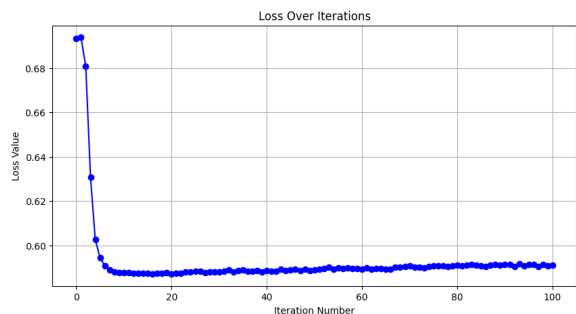


Figure 21: Model Stealing Loss, with selected queries, $\alpha = 0.7$ to 0.5

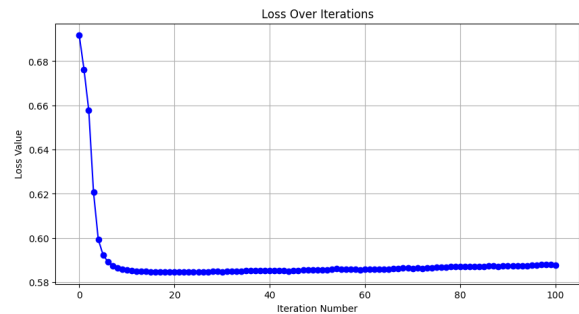


Figure 22: Model Stealing Loss, with random queries, $\alpha = 0.7$ to 0.5