DHBW Mannheim

Bonusabgabe

Nachahmung neuronaler Netze

Bonuseinreichtung: Advanced Machine Learning

Dozent: Prof. Dr. Maximilian Scherer

Vorlesung: Advanced Machine Learning

Kursnummer: WWI-18-DSA

Name: Bulkens, Björn

Matr.-Nr.: 9936663

E-Mail: s182146@student.dhbw-mannheim.de

Fachsemester: 06

Studiengang: BA Wirtschaftsinformatik Data Science

Abgabedatum: 22. Juli 2021

Inhaltsverzeichnis

1 Worum geht es

In dieser kurzen Arbeit geht es um das *Nachahmen* bereits bestehender Modelle. *Nachahmen* heißt dabei, dass es bereits ein Modell gibt das auf einem zuvor definierten Problem arbeitet bzw. für dieses trainiert wurde, jedoch ein *Angreifer* dieses für sich selbst nutzen möchte ohne eventuelle Lizenzkosten oder ähnliches für die Verwendung zu zahlen. Es ist prinzipiell über einen API-Zugriff, ohne wissen über Architektur oder ähnlichen Information, möglich ein sehr ähnliches Modell nachzubauen. Dieses Vorgehen bezeichnet man als Adversarial Extraction Attack und soll im folgenden näher behandelt werden.

2 Einführung in Adversarial Attacks

Da das Thema dieser Arbeit in den Teilbereich der Adversarial Attacks fällt soll als erste Einführung das Konzept und die Theorie dieser kurz erklärt werden. Vorweg als Definition, die dann im Verlauf der Erklärung näher erläutert wird: Adversarial Attacks bezeichnen Methoden, mit denen Adversarial Examples erzeugt werden, um ein Netz bewusst in die Irre zu führen und somit vermeintlich Schaden am System anrichten (Chakraborty et al., 2018) . Der Schaden wäre in diesem Fall finanziell in Form von Umsatz und unternehmerisch in Verlust von einem Vorteil bzw. Wissen. Es wurde unter Umständen viel Zeit und Geld darin investiert die richtige Datenquelle, Architektur und Hyperparameter für das Modell zu finden und anzupassen.

2.1 Arten von Adversarial Attacks

Es wird grundsätzlich in drei übergeordnete Kategorien von Adversarial Attacks unterschieden, die unterschiedliche Ziele verfolgen und eigene Methoden und Vorgehensweisen haben:

- *Poisoning Attack* - Es findet eine Beeinflussung des Modells über eine Manipulation der Traingsdaten und somit über den Trainingsprozess selbst statt. „Ein Angreifer versucht die Traingsdaten zu vergiften durch das Hinzufügen sorgfältig entworfener Beispiele, um letztendlich den gesamten Trainingsprozess zu beeinflussen“ (Chakraborty et al., 2018, S. 5) .

- *Evasion Attack* - Hierbei findet die Beeinflussung nach Inbetriebnahme des ursprünglichen Modells statt (Chakraborty et al., 2018, S. 5) . Der Angreifer versucht das Modell bewusst zu beeinflussen oder schlicht zu Fehlklassifikation zu bringen.

Dieser Prozess stellt meist ein eigenes Optimierungsproblem dar und können nochmal in ihre unterschiedlichen Vorgehensweisen unterteilt werden (Ibitoye et al., 2019) Diese sind jedoch für den weiteren Verlauf der Arbeit nicht relevant.

- *Oracle Attack* - Bei der der Angreifer über einen Blackbox-Zugriff versucht, so viel wie möglich über das Modell und seine verwendeten Daten herauszufinden. Ebenfalls ohne das Training zu beeinflussen (vgl. Chakraborty et al., 2018, S. 6.) . Es werden Anfragen an das Modell gesendet und auf Basis der zurück erhaltenen Information, genauer der Klassifikation mit/ohne deren Wahrscheinlichkeit, wird beispielsweise versucht, ein eigenes Netz aufzubauen oder andere Informationen bezüglich der Entscheidungsgrenzen zu extrahieren. Hinzu kommt eine dadurch entstehende Gefahr für Datenschutz und Privatsphäre, da der Angreifer bei einer sogenannten *Inversion Attack* versuchen kann Trainingsdaten wiederherzustellen (Ibitoye et al., 2019; Fredrikson et al., 2015) oder das hier näher behandelte Thema der Nachahmung.

3 Extraction Attacks

Extraction Attacks sind Teil der Gruppe der *Orcale Attacks*; Ihr Ziel ist es Details über die Architektur des angegriffenen Modells abzuleiten (Ibitoye et al., 2019) . Hierzu zählen Aufbau, Parameter und Gewichte. Bekannte Werke zu diesem Teilgebiet sind: Correia-Silva et al., 2018, Jagielski et al., 2020 und Orekondy et al., 2019, neben weiteren.

3.1 Copycat CNN

Das Copycat-Verfahren für CNNs nach (Correia-Silva et al., 2018) ist ein solches Verfahren Modelle, hierbei speziell CNNs, nachzuahmen in ihrer Performance. Dabei wurde spezieller noch die These überprüft, ob hierfür hierfür Bilder aus der Problem-Domain notwendig sind. Die Problem-Domain ist dabei der Bereich der das Problem umfasst, wie beispielsweise Bilder von Gesichtern beim Training einer Gesichtserkennung oder ähnliches.
Dies ist in sofern relevant, als dass der Erhalt von Daten aus der Problem-Domain sehr kostspielig sein kann, beispielsweise hochauflösende Bilder von Bakterien aus Elektronenmikroskopen oder andersherum Bilder von Planten aus den Teleskopen großer Sternwarten. Diese stehen dem Angreifer unter Umständen nicht zur Verfügung und bilden die Daseinsberechtigung für Lizenzgebühren oder ähnliches zur Nutzung des trainierten Modells. Dem Angreifer würde demnach nur bleiben andere natürliche Bilder

für das Training seiner *Copycat* zu verwenden.
Correia-Silva et al., 2018 basieren ihre These, dass ein solches Vorgehen möglich ist, auf den *Erfolg* von Adversarial Attacks und die Anfälligkeit durch CNNs gegenüber diesen. Sie leiten daraus eben jene unabhängig von der Problem Domain ab, was ihrer Meinung das Feld des Transfer Learnings weiter voranbringen kann.

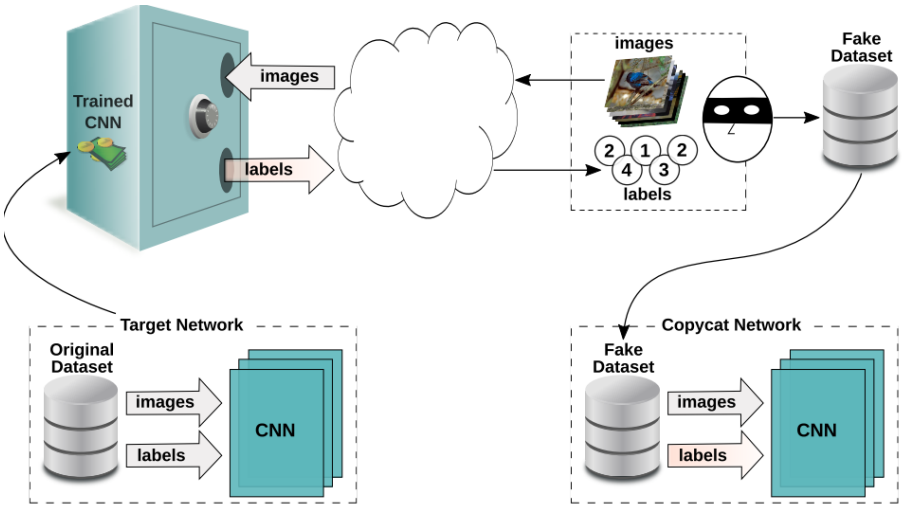

Abbildung 1: Veranschaulichung des Prozesses bei einem Copycat-Angriff aus Correia-Silva et al., 2018

Das Vorgehen ist dabei wie folgt und in Abb. 2 visualisiert: Zuerst wird ein Menge an Daten, wahlweise aus oder auch nicht aus der Problem-Domain durch den Angreifer gesammelt, an das Original-Modell weitergegeben und ein gelabelter *Fake-Datensatz* aus jenen Inputs und den zurückerhaltenden Outputs erstellt. Wichtig ist dabei zu beachten, dass eventuelle Fehlklassifikation (wenn man auf der PD trainiert) übernommen werden. Mit diesem neuen Datensatz kann dann ein eigenes Modell trainiert werden.
Da es Best-Practise ist seine Modelle auf vortrainierten Modellen und etablierten Modellarchitekturen zu basieren, gilt es bei der Auswahl der Copycat abzuschätzen was eine geeignete Architektur für das Problem wäre, kann diese wählen und bespielsweise noch den letzten Layer auf die Anzahl der gewünschten Outputs anpassen. Das vortrainierte Modell mit seinen Gewichten kann nun für dieses ungewöhnliche Transfer Learing genutzt und mit Hilfe des Fake-Datensatz weiter trainiert werden.

3.2 Knock Off Nets

Das Knockoff-Verfahren nach (Orekondy et al., 2019) ist ein alternativer Ansatz an das Nachahmen der Funktionalität des Modells. Ziel ist auch hier nicht das Nachbauen der Architektur oder der Parameter, sondern die bloße Funktionalität des zu stehlenden Netzwerks.

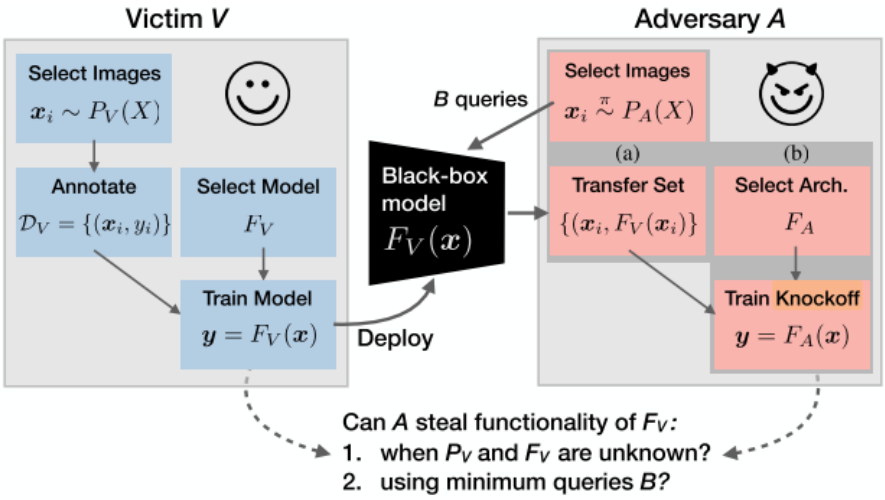

Abbildung 2: Veranschaulichung des Prozesses bei einem Knockoff-Angriff aus Orekondy et al., 2019

Genau wie beim vorherigen Copycat-Angriff geht man davon aus, dass das Opfer (Victim „V“) seinen Klassifikator mit speziellen Daten trainiert hat, die dem Angreifer nicht zur Verfügung stehen. Er bildet aus seinen speziellen Daten einen annotierten Datensatz, trainiert auf diesem seinen Klassifikator und stellt es dann als API oder ähnliches zur Verfügung. Der Angreifer auf der anderen Seite hat keinen Zugriff auf das Modell und kann einfache Anfragen an das Modell schicken und Klassifikationen zurückerhalten. Es sind Blackbox üblich keine Architektur, Parameter oder ähnliches bekannt, sondern nur das trainierte Problem und die zurückerhalten Klassifikationen.
Weiterhin parallel zur Copycat-Methode nutzen wir einen eigenen ungelabelten Datensatz, denn der Angreifer durch die Blackbox des V's für sich klassifizieren lässt. Gehen wir dabei beispielsweise von einer bezahlten API aus besteht ein monetäres Interesse des Angreifers darin, die Funktionalität in möglichst wenig Anfragen übernehmen zu können. Darauf aufbauend findet ein vollständiger Reinforcement Learning Ansatz statt, um möglichst Effizient Anfragen an die Blackbox schicken zu können und möglichst schnell und in wenig Anfragen die Leistung des Opfers möglichst gut nachahmen zu können (Orekondy et al., 2019) . Hierauf soll aber nicht näher eingegangen werden, da letztendlich im Zuge dieses Projektes nur ein vereinfachter Copycat-Ansatz umgesetzt wurde.

4 Umsetzung

Für die Umsetzung des Projekts wurde zum einen Ansatz über die Implementation der Adversarial Robustness Toolbox von Trusted-AI (Nicolae et al., 2018) als auch eine

eigene kleine Exploration des Copycat-Ansatzes umgesetzt.
Die Implementierung über das Framework scheiterte an den Übersetzungen zwischen eigenen Modellen und Framework, weshalb zumindest der Kerngedanke der Copycat-CNNs: CNNs können auch gut mit Bildern außerhalb der Problemdomain trainiert werden, wenn diese durch ein für das gewünschte Problem geeignete Netz gelabelt werden. Zu dieser These wurde das trainierte ResNet eines Kommulitonen, welches Bienen von Ameisen unterscheiden hergenommen und für die Bildung eines Fake-Datensatz auf der Basis des CIFAR-100-Datensatz genutzt. Auf diesem neuen Fake-Datensatz wurde dann, entsprechend der These von Correia-Silva et al., 2018 auf einer üblichen Modellarchitektur trainiert, hier VGG. Zwar erreichte das Copycat-Netzwerk bereits 20 Epochen eine Validation Accouracy von 92% auf dem Fake-Datensatz, dennoch war es bei einer Evaluierung auf dem realen Datensatz aus der Problemdomaine nicht besser als eine zufällige Klassifizierung. Dies wurde auch über mehr Epochen hinweg nicht gesteigert.
Die Verwendung eines eigenen ResNets, das auf die selbe Weise wie das VGG zuvor trainiert wurde, zeigte hier jedoch ähnliche Ergebnisse, was auf andere Fehlerquellen hinweist als die Netzarchitektur. Weiteres Parameter-Tuning und Fehleranalyse ergaben, dass der Victim-Classifier auf dem CIFAR-Datensatz überproportial oft Ameisen vorhersagt und entsprechend auch die Copycat dieses Verhalten möglichst nachahmte. Hier stellt sich eine klare Schwäche der These heraus: Auch wenn Bilder nicht unbedingt aus der Problemdomaine stammen müssen, so müssen sie doch auf eine Art und Weise möglichst vollständig den latenten Raum des ursprünglichen Modells abdecken, da sonst gewisse Bereiche der Entscheidungsgrenze dem Nachahmer völlig unbekannt sind. Die binäre Klassifikation eignet sich hierbei zum Visualieren dieses Problems sehr gut, da es sich direkt in einer Fehlklassifikation der Hälfte der Daten ausdrückt, wenn die dafür nötigen Entscheidungsgrenzen nicht sorgfältig erforscht wurden, wohingegen ein solcher Umstand bei beispielsweise 100 Klassen leicht untergehen kann.
Auch sei gesagt, dass die eigene Implementierung natürlich nur ein deutlich vereinfachter Ansatz der eigentlichen Copycat-Methode nach Correia-Silva et al., 2018 ist und die Authoren dort sehr hohe Genauigkeiten durch eine gute Wahl des Datensatzes und andere Techniken erzielen konnten, auch ohne Bilder aus Problemdomaine zu haben. Dieser Ansatz diente mehr der Veranschaulichung der grundsätzlichen Idee und ist auf Github unter https://github.com/FatManWalking/ml_bonus/blob/main/Assignement_9936663.ipynb zu finden.

Literatur

Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A. & Mukhopadhyay, D. (2018). Adversarial Attacks and Defences: A Survey. http://arxiv.org/abs/1810.00069

Correia-Silva, J. R., Berriel, R. F., Badue, C., De Souza, A. F. & Oliveira-Santos, T. (2018). Copycat CNN: Stealing Knowledge by Persuading Confession with Random Non-Labeled Data. *Proceedings of the International Joint Conference on Neural Networks*, *2018-July*. https://doi.org/10.1109/IJCNN.2018.8489592

Fredrikson, M., Jha, S. & Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. *Proceedings of the ACM Conference on Computer and Communications Security*, *2015-Octob*, 1322–1333. https://doi.org/10.1145/2810103.2813677

Ibitoye, O., Abou-Khamis, R., Matrawy, A. & Omair Shafiq, M. (2019). *The threat of adversarial attacks on machine learning in network security - A survey* (Techn. Ber.).

Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A. & Papernot, N. (2020). High accuracy and high fidelity extraction of neural networks. *Proceedings of the 29th USENIX Security Symposium*, 1345–1362.

Nicolae, M.-I., Sinn, M., Tran, M. N., Buesser, B., Rawat, A., Wistuba, M., Zantedeschi, V., Baracaldo, N., Chen, B., Ludwig, H., Molloy, I. & Edwards, B. (2018). Adversarial Robustness Toolbox v1.2.0. *CoRR*, *1807.01069*. https://arxiv.org/pdf/1807.01069

Orekondy, T., Schiele, B. & Fritz, M. (2019). Knockoff nets: Stealing functionality of black-box models. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*, 4949–4958. https://doi.org/10.1109/CVPR.2019.00509

Eigenständigkeitserklärung

Hiermit versicheren ich, dass ich die Hausarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, alle Ausführungen, die anderen Schriften wörtlich oder sinngemäß entnommen wurden, kenntlich gemacht sind und die Arbeit in gleicher oder ähnlicher Fassung noch nicht Bestandteil einer Studien- oder Prüfungsleistung war.

Mannheim, den 22. Juli 2021 _____

Björn Bulkens